НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА СИСТЕМНОГО ПРОГРАМУВАННЯ І
СПЕЦІАЛІЗОВАНИХ КОМП'ЮТЕРНИХ СИСТЕМ

«На правах рукопису»
УДК 004

«До захисту допущено»

Завідувач кафедри СПСКС

_____ В.П.Тарасенко_
(підпис) (ініціали, прізвище)

"___"_____2018р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 123 Комп'ютерна інженерія (Системне програмування)

на тему ЗАСОБИ МОНІТОРИНГУ ЕНЕРГОЕФЕКТИВНИХ ТА ВІДНОВЛЮВАЛЬНИХ ДЖЕРЕЛ ЕНЕРГІЇ

Виконав: студент ІІ курсу, групи КВ-72мп
(шифр групи)

Мацегора Михайло Віталійович
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник к.т.н., доцент Тарасенко-Клятченко О.В.
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент_____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2018 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп‘ютерних систем

Рівень вищої освіти – другий (магістерський) за освітньо-професійною програмою

Спеціальність (спеціалізація) – 123 Комп’ютерна інженерія (Системне програмування)

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ В. П. Тарасенко

«___»_____2018 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Мацегорі Михайлу Віталійовичу

1. Тема дисертації «Засіб моніторингу енергоефективних та відновлювальних джерел енергії», науковий керівник дисертації Тарасенко-Клятченко Оксана Володимирівна, к.т.н., доцент, затверджені наказом по університету від «30» жовтня 2018 р. № 4030-с

2. Термін подання студентом дисертації «__» грудня 2018 р.

3. Об’єктом дослідження є засіб моніторингу у вигляді веб-орієнтованої системи, який забезпечить користувача повним обсягом інформації щодо відновлювальних джерел енергії по введеним ним критеріям пошуку.

4. Предметом дослідження виступають веб-технології та засоби створення веб-орієнтованої системи для зберігання та відстежування актуальних відомостей щодо відновлювальних джерел енергії.

5. Перелік завдань, які потрібно розробити:

- провести аналіз існуючих засобів моніторингу;
- розробити клієнт-серверну архітектуру веб-орієнтованої системи;

– створення необхідного програмного забезпечення для демонстрації роботи засобу моніторингу;
– проаналізувати результати роботи системи.

6. Перелік ілюстративного матеріалу: презентація (кількість аркушів: 10)
7. Орієнтовний перелік публікацій:
   – Тези доповіді «Прикладна математика та комп'ютинг» ПМК-2018-2
   – Тези доповіді на V-й Міжнародній науково-технічній Internet-конференції НУХТ, АКС 2018
8. Дата видачі завдання 04 жовтня 2017 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Ґрунтовне ознайомлення з предметною галуззю	25.10.2017	
2.	Визначення структури магістерської дисертації; вивчення літератури, пошук додаткової літератури, патентний пошук	17.03.2018	
3.	Робота над першим розділом магістерської дисертації; проведення наукового дослідження	25.04.2018	
4.	Проведення наукового дослідження; робота над другим розділом магістерської дисертації; розроблення програмного забезпечення	24.05.2018	
5.	Проведення наукового дослідження; робота над статтею за результатами наукового дослідження	16.06.2018	
6.	Проведення наукового дослідження; робота над третім розділом магістерської дисертації	17.07.2018	
7.	Завершення роботи над основною частиною магістерської дисертації; підготовка матеріалів доповіді на конференції ПМК-2018	21.10.2018	
8.	Оформлення текстової і графічної частини магістерської дисертації	15.11.2018	

Студент М. В. Мацегора

Науковий керівник дисертації О. В. Тарасенко-Клятченко

РЕФЕРАТ

Актуальність теми.

Проблема енергозбереження в Україні є досить актуальною на сьогоднішній день, оскільки недостатня забезпеченість власними енергоресурсами та неефективне їх використання зумовлюють зниження рівня національної економіки країни і підвищення її енергетичної залежності від інших країн. Тому, поступово відбувається впровадження альтернативних джерел енергії в наше повсякденне життя.

Об'єктом дослідження є засіб моніторингу у вигляді веб-орієнтованої системи, який забезпечить користувача повним обсягом інформації щодо відновлювальних джерел енергії по введеним ним критеріям пошуку.

Предметом дослідження виступають веб-технології та засоби створення веб-орієнтованої системи для зберігання та відстежування актуальних відомостей щодо відновлювальних джерел енергій.

Мета роботи: створення засобу моніторингу енергоефективних та відновлювальних джерел енергії.

Методи дослідження. В роботі використовуються методи інтелектуального аналізу даних для автоматичного пошуку та обробки інформації.

Наукова новизна роботи полягає в наступному:

1. Розроблено засіб моніторингу у вигляді веб-орієнтованої системи з інтерактивною картою енергоефективних та відновлювальних джерел енергії, який допоможе у реальному часі підібрати користувачу те джерело енергії, яке підійде саме для нього по його параметрам.

Практична цінність отриманих в роботі результатів полягає в тому, що запропоновані засоби дають змогу підібрати користувачам відповідне відновлювальне джерело енергії, враховуючи усі фактори впливу.

Апробація роботи. Запропонований засіб був представлений та обговорений на науковій конференції магістрантів та аспірантів «Прикладна математика та комп'ютинг» ПМК-2018 (Київ, 14 - 16 листопада 2018 р.) та на

V Міжнародній науково-технічній Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами», яка проводилась 22 листопада 2018 р. у Національному університеті харчових технологій.

Структура та обсяг роботи. Магістерська дисертація складається з вступу, трьох розділів, висновків та додатків.

У вступі надано загальну характеристику роботи, виконано оцінку сучасного стану проблеми, обґрунтовано актуальність напрямку досліджень, сформульовано мету і задачі досліджень, показано наукову новизну отриманих результатів і практичну цінність роботи, наведено відомості про апробацію результатів і їх впровадження.

У першому розділі розглянуто основні способи створення та аналіз засобів моніторингу.

У другому розділі були розглянуті веб-технології та засоби створення веб-орієнтованої системи у вигляді інтерактивної карти.

У третьому розділі запропоновано програмну реалізацію веб-орієнтованої системи.

У висновках проаналізовано отримані результати роботи.

У додатках наведено алгоритм створення засобу моніторингу енергоефективних та відновлювальних джерел енергії.

Робота виконана на 79 аркушах, містить 4 додатків та посилання на список використаних літературних джерел.

Ключові слова: обробка та аналіз інформації, інтерактивна карта, веб-орієнтована система, моніторинг

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

СМ – система моніторингу.

ОВ – обчислювальні величини.

CSS – каскадні таблиці стилів.

СУБД – система управління базами даних.

Скрипт - це програма або програмний файл сценарій, які автоматизують деяку задачу, яку користувач робив би вручну, використовуючи інтерфейс програми.

Двіжок веб-системи - система управління контентом сайту.

CMS - система управління контентом сайту.

PHP - скриптова мова загального призначення, інтенсивно застосовується для розробки веб-додатків.

API – опис способів, якими одна комп'ютерна програма може взаємодіяти з іншою програмою.

MVC (Model – View – Controller) - схема розділення даних програми, інтерфейсу користувача та логіки управління на три окремі компоненти: модель, вид та контролер - так що кожен компонент може бути модифікований самостійно.

HTML - мова гіпертекстової розмітки.

Frontend - клієнтська сторона призначеного для користувача інтерфейсу до програмно-апаратної частини сервісу.

Backend - програмно-апаратна частина сервісу.

LESS - це надбудова над CSS.

JQuery - бібліотека JavaScript, що фокусується на взаємодії JavaScript і HTML.

SLA - угода про рівень послуг.

SNMP - стандартний інтернет-протокол для управління пристроями в IP-мережах

HTTP - протокол прикладного рівня передачі даних.

AJAX - технологія звернення до сервера без перезавантаження сторінки.

View - уявлення, код якого необхідно обробити сервером додатка.

БД – база даних.

ВП - веб-приложение.

ВСТУП

Проблема енергозбереження в Україні є досить актуальною на сьогодні, оскільки недостатня забезпеченість власними енергоресурсами та неефективне їх використання зумовлюють зниження рівня національної економіки країни і підвищення її енергетичної залежності від інших країн. Тому поступово відбувається впровадження альтернативних джерел енергій в наше повсякденне життя.

До альтернативних джерел енергії належать відновлювальні – вітер, сонячне випромінювання, енергія морів і океанів. Їх перевагою є те, що всі вони екологічно чисті. На відміну від викопних палив ці форми енергії не обмежені геологічно накопиченими запасами. Це означає, що їх використання і споживання не веде до неминучого вичерпання запасів.

Більшість людей, щоб встановити у своїх домівках відновлювальні джерела енергії, звертаються до енергокомпаній за консультацією і витрачають немалі гроші за цю послугу . І тому, я вирішив створити засіб моніторингу у вигляді веб-орієнтованої системи з інтерактивної картою, яка у реальному часі допоможе користувачу підібрати, яке джерело енергії підійде саме для нього по його параметрам. Адже вибір певного джерела енергій залежить від багатьох факторів, які будуть враховуватись в веб-орієнтованій системі.

Архітектура засобу моніторингу у вигляді веб-орієнтованої системи була виконана на основі самописної платформи BlizAnt зі зручною оболонкою для створення веб-застосунків та сайтів і управління їхнім контентом. Вона інкапсулює обробку об'єктів додатків високого рівня, таких як, наприклад, запит і відповідь, що робить їх доступними для розробника.

Для того, щоб структурувати серверну частину веб-орієнтованої системи, були взяті принципи архітектурного рішення MVC.

Принцип MVC дозволяє розділити реалізацію логіки докладання, зовнішній вигляд (графічний інтерфейс, GUI) і взаємодія з користувачем.

Це призводить до більш структурованого коду, спрощує підтримку коду, робить його більш логічним і зрозумілим.

1. АНАЛІЗ ТА СПОСОБИ СТВОРЕННЯ ЗАСОБІВ МОНІТОРИНГУ

Моніторинг - система постійного спостереження за явищами і процесами, що проходять в навколишньому середовищі і суспільстві, результати якого служать для обгрунтування управлінських рішень по забезпеченню безпеки людей та об'єктів економіки.

При автоматичному контролі відбувається отримання і обробка інформації про стан об'єкта і зовнішніх умов для виявлення подій, що визначають управлінські дії. Подією може бути будь-який якісний результат: поява деталі з розмірами, що виходять за допустимі межі, коротке замикання, вихід температури за встановлене значення, аварія обладнання та інші.

1.1. Класифікація засобів моніторингу

Системи моніторингу, найважливіша складова управління об'єктами, вимагає для свого визначення, становлення і вивчення наукового підходу, важливим інструментом якого є класифікація. Класифікація являє собою спосіб упорядкування, встановлення зв'язків і відношень між властивостями і функціями СМ. У процесі класифікації визначаються класи - групи об'єктів, що мають визначені загальні ознаки, що забезпечують їхнє зіставлення й ідентифікацію, шляхом установлення приналежності до визначеного класу. Визначення класифікації має і прикладне значення, тому що визначає спільність принципових технологічних і функціональних методів дослідження, розробки, побудови і забезпечення функціонування систем і об'єктів моніторингу.

Можна розглядати окремо класифікацію об'єктів моніторингу і класифікацію систем моніторингу. Ці класифікації природно будуть тісно взаємозалежні, тому обмежимося розглядом класифікації СМ.

У якості об'єкта моніторингу може виступати люба система або процес. По суті справи спостереженню підлягають не статичні або повільно мінливі характеристики об'єктів, а процеси, що відбуваються на цих об'єктах, і їх

показники, що їх характеризують. Тому, коли мова йде про об'єкт моніторингу, вірніше говорити не про об'єкти, на яких виконується моніторинг, а про процеси, що відбуваються на цих об'єктах. Коли говорять, що вона забезпечує об'єкт, то мається на увазі, що СМ обслуговує процес функціонування об'єкта. Може застосовуватися цілий комплекс або багатоканальні СМ для забезпечення функціонування багатьох процесів, що відбуваються на великому об'єкті. Об'єкт має власні характеристики, але вони звичайно мають статичний характер і їх визначення не входить в область обслуговування СМ. Тому в даному випадку найменування об'єкта (наприклад назва промислового підприємства) лише ідентифікує ті процеси, що являють собою суб'єкт або суб'єкти моніторингу.

Найбільш зручною і наглядною є строго ієрархічна класифікація, але властива їй заборона на перетинання окремих класів не дозволяє її використовувати в багатьох випадках. Тому ми скористаємося не строго ієрархічною класифікацією, що допускає перетинання класів, але має властивість строго ієрархічної класифікації: кожний підклас має попередній клас, що може бути не один (як у строго ієрархічній класифікації), але проте відношення підпорядкованості виражається цілком чітко.

Розподіл СМ по класах здійснюється на основі найбільш характерних властивостей (ознак), що властиві СМ. Такими ознаками є:

- предметні області призначення СМ;
- масштабність СМ;
- характер відношення об'єктів моніторингу до зовнішнього середовища;
- функціональна приналежність об'єктів, що спостерігаються;
- дисципліна дискретності спостереження.

Системи моніторингу великомасштабних об'єктів мають досить високу складність, реалізують функції моніторингу і обробки моніторингової інформації, що потребує високого інтелектуального рівня і повинні забезпечуватися сучасними інформаційними комп'ютерними технологіями,

які реалізують автоматизацію задач функціонування СМ. Вся множина задач, що автоматизуються та забезпечують здійснення функцій СМ, повинна підтримувати процеси діяльності, які сукупно складають інформаційну технологію моніторингу: стеження - оцінка - прогноз - рішення.

Найважливішою характеристикою (критерієм оцінки якості) будь-якої системи моніторингу є дискретність спостереження ВВ. Зменшення періоду дискретності веде до збільшення надмірності інформації, одержуваної в результаті моніторингу, що веде до зайвого перевантаження каналів передачі інформації, систем збереження й обробки інформації. Невиправдане збільшення періоду дискретності веде до втрати точності визначення результатів і підвищенню рівня невизначенності результатів моніторингу. Тому період дискретності повинний бути обгрунтованим.

Серед критеріїв оцінки СМ дискретність займає особливе місце. Це пояснюється не тільки тим, що від неї залежить точність і складність побудови і забезпечення функціонування СМ. Цей критерій є основним показником для узгодження СМ з об'єктом спостереження, споживачами інформації СМ, пристроями перетворення інформації, що можуть входити до складу СМ або знаходиться за її межами.

Неоднаковість ВВ і умов їх формування визначають необхідність встановлення індивідуального періоду дискретності для кожної окремої ВВ. Однак, в умовах, коли число ВВ досягає сотень і тисяч, організація використання індивідуальної періодичності для кожної ВВ стає технічно складною. Тому намагаються укомплектувати ВВ у групи з однаковою періодичністю спостереження. Найбільш прості для реалізації технології забезпечують постійну дискретність для всіх ВВ.

Сфери використовування моніторингу надзвичайно різноманітні. Численні системи моніторингу володіють деякими загальними характеристиками, що дає можливість говорити про моніторинг як цілісний самостійний науково-практичний феномен. Відмінності ж в тлумаченні суті моніторингу, в цілеукладання і засобах його здійснення відображають

специфіку і рівень розробленості проблем моніторингу в кожній з областей його застосування.

Існування великої кількості різних систем моніторингу породжує необхідність їх певного впорядкування. З цією метою ми зробили спробу класифікації існуючих систем моніторингу за декількома підставами. Як одну з таких підстав можна розглядати область вживання моніторингу. Це дозволяє виділити наступні його види:

- в екології і біології: моніторинг повітря, води, лісів, рівня моря, повітря, кліматичної системи, клімату, температури, оточуваного середовища, сейсмологічний моніторинг, токсичних газів, шуму, випромінювання, екологічний, грунтово-хімічний, переселення пташиного населення, здоров'я тварин, і інші.
- у медицині: санітарно-гігієнічний, медичний, хворих на рак, внутріутробного розвитку зародка, температури, аритмії, серцевої діяльності, кров'яного тиску під час анестезії, глюкози в крові і ін.
- у економіці і бізнесі: сільськогосподарської продукції, цін, бізнесу, податків, устаткування, доходів, ринку праці, ринку продуктів харчування, будівельних товарів, цін на ГКО і ін.
- у політиці, політології і соціології: засобів масової інформації, регіональних ЗМІ, виборів, прав людини, новин ТБ, соціально-політичний моніторинг регіонів, Російського законодавства, поточного законодавства, економічного законодавства, соціально-економічної ситуації.
- у промисловості: корозії металів, промислових, каталітичних процесів.
- комп'ютерів і засобів зв'язку: моніторинг мереж, радіокоммунікацій, комп'ютерних систем, короткохвильових радіопередач, надійності даних і ін.
- у освіті: знань систем і ін., що вчаться, освітніх

Другою підставою для класифікації систем моніторингу можуть бути засоби, що використовуються для його проведення. На цій підставі можна виділити, авіаційний, космічний, дистанційний, супутниковий, інструментальний, педагогічний, психологічний, соціологічний, медичний, статистичний види моніторингу. Дані визначення відносяться в своїй більшості до систем моніторингу і, в якій то мірі, відображають його розвиненість, ступінь і рівень інструментування.

У якості третьої підстави для класифікації систем моніторингу можна запропонувати способи збору інформації, що використовуються. На підставі цього існуючі системи моніторингу можна підрозділити на чотири групи:

- До першої групи можна віднести ті види моніторингу, в процесі здійснення якого можливий безпосередній опис об'єкту моніторингу, не вдаючись до яких-небудь вимірювань, використовуючи технології структуризації результатів, побудову схеми і технології збору інформації (наприклад, моніторинг засобів масової інформації, поточного законодавства, виборів).
- Другу групу складають види моніторингу, в процесі якого здійснюється безпосереднє фізичне вимірювання параметрів об'єкту (наприклад, моніторинг шуму, рівня моря, податків, корозії металів, комп'ютерних мереж, ринку продуктів).
- Третя група включає види моніторингу, в ході якого вимірювання параметрів об'єкту проводиться з використанням системи добре розроблених і загальноприйнятих критеріїв або індикаторів (наприклад, моніторинг повітря, серцевої діяльності, доходів, грунтово-хімічний моніторинг).
- Четверту групу складають ті види моніторингу, в процесі якого вимірювання проводиться опосередковано, із залученням технологій наукового дослідження, з використанням системи критеріїв і показників (наприклад, моніторинг санітарно-

гігієнічний, соціально-політичний, соціально-економічної ситуації).

Окрім цього, існуючі системи моніторингу можна розділити на групи у відповідності з їх орієнтацією на конкретного користувача. В рамках кожної з груп розв'язуються проблеми уявлення і розповсюдження інформації, одержуваної в процесі моніторингу, а також проблеми оплати його організації і проведення.

Можна виділити три групи, відмінні по кількості користувачів і інтенсивності використовування результатів моніторингу відповідним користувачем:

- Першу групу складають види моніторингу орієнтовані на суспільство в цілому. Метою такого моніторингу може бути, наприклад, формування громадської думки. Види моніторингу, результати яких призначені для такого роду аудиторії нечисленні. Ознайомлення користувача з результатами моніторингу в цьому випадку здійснюється через засоби масової інформації, у тому числі і електронні. Як правило, оплата такого роду моніторингу проводиться за допомогою системи бюджетного фінансування.
- Друга група включає види моніторингу, орієнтованого на фахівців відповідних областей діяльності. Це, вважається, найчисленніша група, До неї належить більшість існуючих систем моніторингу. При цьому, самі групи фахівців, для яких призначені результати кожного конкретного моніторингу можуть бути як достатньо малі, так і дуже численні. Основними способами розповсюдження одержуваної в ході такого виду моніторингу інформації є спеціалізовані видання, зокрема періодичні, ІНТЕРНЕТ, підписка. Оплата цього виду моніторингу проводиться користувачами, причому кожний користувач оплачує тільки частину витрат.
- Третя група включає види моніторингу, користувачами якого є конкретні органи управління, керівники, окремі структури. В

літературі мало представлені види моніторингу, що входять до цієї групи, проте сам жанр друкарського видання має на увазі достатньо масове використання. Цілий ряд фірм пропонує і реалізує цільові моніторинги, користувачами яких є виключно керівники. Засобом розповсюдження інформації, одержуваної в ході такого роду моніторингу є аналітичні звіти, рекомендації, проекти, які як правило не мають широкого розповсюдження. В цьому випадку оплата робіт проводиться як правило тільки замовником.

Також, можна виділити два типи моніторингу, перший з яких направлений на реалізацію задач функціонування, а другий - задач розвитку. Інакше кажучи, одні системи моніторингу, виконавши свою конкретну задачу, припиняють своє існування, інші можуть існувати необмежено довго. Вони можуть здійснюватися протягом не одного десятиліття або навіть сторіччя (наприклад, нагляди за погодою). Причини завершення функціонування тієї або іншої системи моніторингу можуть бути двоякого роду:

- сам об'єкт моніторингу може припинити своє існування;
- об'єкт моніторингу перестає бути небезпечним (приклади такого роду об'єктів - рівень моря, в тому випадку, якщо він достатньо довго залишається стабільним, вибори, після їх завершення і аналізу результатів.).

Якщо проаналізувати характер можливих об'єктів моніторингу можна відзначити, що ними можуть бути як складні системні об'єкти (наприклад, здоров'я, клімат, екологічний стан, рівень економіки, засоби масової інформації, радіокомунікації, ціни і ін.), так і достатньо локальні (наприклад, переселення птахів, регіональні вибори, якість роботи економічних або конкретних мереж, кров'яний тиск під час анестезії і ін.). Проте, є щось загальне, що об'єднує всі ці різнорідні об'єкти, що належать різним сферам діяльності.

Можна виділити дві основні особливості об'єктів моніторингу.

Перша з них - це їх динамічність. Всі об'єкти, вивчення або обстеження яких здійснюється із застосуванням моніторингу знаходяться в постійній зміні, розвитку.

Друга особливість - це наявність або можливість небезпеки, що виникає в процесі функціонування об'єкту моніторингу.

Задачею моніторингу є попередження про те або інше неблагополуччя, небезпеки, в широкому розумінні цього слова, для ефективного функціонування об'єкту. Причому не просто констатація факту появи змін, що представляють небезпеку, а саме попередження про неї до того як ситуація може стати необоротною. Тим самим створюється можливість запобігти або мінімізувати можливий деструктивний розвиток подій.

Динамічність об'єкту, можливість виникнення небезпеки в процесі його функціонування і розміри небезпеки визначають необхідність і доцільність використання моніторингу для дослідження, а також вибір тієї або іншої конкретної системи моніторингу.

Окрім цього необхідно наголосити і на ще одній особливості, що розповсюджується правда не на всі перераховані види моніторингу - можливість побудови прогнозу розвитку тієї або іншої системи в умовах відсутності флуктуаційних відхилень або форс-мажорних обставин, що додає моніторингу особливу цінність і значущість з погляду потенційного користувача.

1.2. Аналіз методів розробки засобів моніторингу обчислювальних мереж

В даний час велике значення має завдання дослідження і розробки методів аналізу даних моніторингу обчислювальних мереж з автоматичним прогнозуванням критичних ситуацій для забезпечення автоматизації процесу прийняття рішення щодо дій у критичних ситуаціях, таких як відмови

підсистем, загрози інформаційній безпеці, перевантаженість оперативної або фізичної пам'яті.

На даному етапі розвитку систем моніторингуособливий інтерес представляють питання, пов'язані з розробкою алгоритмічного і математичного забезпечення систем моніторингу з впровадженням інтелектуальних технологій аналізу даних, розширення функціональних можливостей систем моніторингу великомасштабних обчислювальних комплексів [1].

Потенціал сучасних обчислювальних систем дуже великий, але досягнення максимальної ефективності їх роботи неможливе без спостереження адміністратором системи за потоком завдань. З ростом обчислювальних систем стає більш складною і завдання визначення ефективності їх роботи. І чим складніше система, чим більше процесорів і ядер вона використовує, тим складніше розуміти її поведінку, визначати вузькі місця і причини зниження продуктивності [2].

Для спостереження за поведінкою системи адміністратор повинен мати можливість збирати кількісні дані про роботу системи і вміти їх аналізувати.

В сучасних обчислювальних системах більшість додатків, вузлів, мережевих і навіть інфраструктурних пристроїв надають велику кількість доступних для спостереження характеристик. На жаль, у існуючих поширених засобів моніторингу є недоліки, які суттєво ускладнюють збір і аналіз цих даних.

Напрацьований математичний і алгоритмічний апарат для моніторингу за станом великомасштабних обчислювальних комплексів дозволяє досить ефективно вирішувати завдання аналізу даних моніторингу в режимі реального часу. Однак для вирішення завдання підвищення ступеня автоматизації процесу прийняття рішень при аналізі даних моніторингу, а також завдання прогнозування критичних ситуацій необхідна швидка і ефективна робота не тільки з вступниками даними, але і з уже накопиченими. Таким чином, питання розробки математичного і алгоритмічного

забезпечення для аналізу даних моніторингу великомасштабних обчислювальних систем залишається відкритим і вимагає залучення нових сучасних підходів до свого рішення, нових математичних і алгоритмічних інструментів [2].

Існуючі системи дозволяють задавати «штатну» поведінку встановленням меж допустимих значень параметрів обчислювальної системи. Цього недостатньо для опису всіх позаштатних ситуацій, і визначення цих кордонів може бути дуже непростим для окремої системи.

Аналіз накопичених даних і потоку задач допомагає адміністратору у визначенні того, наскільки повно і якісно використовується його обчислювальна система. Для великих систем зберігання всіх даних, що збираються з максимальною точністю недоцільно, а й простого набору базових характеристик недостатньо для аналізу нештатних ситуацій.

До того ж перед користувачами і адміністраторами обчислювальної системи постає ще одна проблема - оптимізоване управління завантаженням системи. Для вирішення даного завдання необхідно реалізовувати функцію прогнозування завантаження, яка дозволить більш ефективно управляти їй за рахунок аналізу вже зібраної статистики по завантаженню в різні періоди часу і при вирішенні різних завдань. Більш того, рішення задачі прогнозування несе в собі і функцію визначення критичних ситуацій, таких як відмови підсистем: електроживлення, охолодження, загрози інформаційної безпеки та інших [3].

Таким чином, однією з найбільш важливих завдань в області моніторингу і аналізу обчислювальних комплексів і комп'ютерних систем є задача розробки методики аналізу ефективності функціонування високопродуктивних обчислювальних систем [1].

Однак єдиного вирішення такого завдання в даний час не існує. Необхідно сформулювати набір вимог для системи моніторингу комп'ютерних мереж, а також для пакета, що дозволяє на їх базі проводити аналіз ефективності роботи комп'ютерної мережі. Іншими словами,

вирішення проблеми моніторингу полягає в розробці та дослідженні алгоритму і його програмної реалізації для дослідження ключових характеристик комп'ютерних систем.

Створюваний програмний комплекс в такому випадку повинен включати як систему управління завданнями і систему моніторингу, так і розвинену систему оповіщення про наявність критичних ситуацій.

Щоб вирішити поставлені завдання, система моніторингу повинна відповідати певним вимогам:

- Масштабованість: комплекс повинен працювати на кластерах з числом процесорів, які змінюються в широкому діапазоні.
- Переносимість: комплекс повинен працювати на платформі Linux з будь-яким дистрибутивом і незначно залежати від особливостей супутнього програмного забезпечення.
- Можливість розширення: комплекс повинен мати можливість відстежити характеристики і параметри, які не передбачені в складі стандартного комплексу, але відображають певні особливості роботи мережі.
- Розподіленість: дані повинні збиратися з фізично розподілених складових частин системи.
- Економічність: робота комплексу не повинна надавати істотного впливу на роботу системи в цілому і призначених для користувача програм, зокрема. Підтримка роботи допоміжних підсистем на обчислювальних вузлах не повинна займати більше 5% процесорного часу, а передача додаткових даних по мережі не повинна перевищувати 1% від загального трафіку.
- Система оповіщення.

Реалізація цих вимог дозволить забезпечити моніторинг комп'ютерної мережі зі зберіганням і аналізом накопичених даних, що дозволяє автономно виявляти нештатні ситуації в системі і оповіщати про це адміністратора.

Пріоритетами розвитку програмного комплексу повинні бути забезпечення масштабованості, запуск в складі діючої обчислювальної мережі в постійному режимі в якості штатного моніторингу, розширення алгоритмів аналізу ефективності роботи мережі в цілому, надання звітів за результатами, а також прогнозування роботи комп'ютерної мережі.

1.3. Порівняльний аналіз систем моніторингу телекомунікаційних мереж

Системи моніторингу існують на ринку телекомунікацій багато років і стрімко розвиваються з розвитком галузі в цілому. Пропоновані на світовому ринку системи моніторингу схожі за виконуваними функціями, всі вони надають майже однаковий мінімальний набір можливостей. Найбільш цікавими є наступні системи моніто-рингу, порівняння яких представлено в таблиці 1.1:

- Argus;
- Intellipool Network Monitor;
- AdRem NetCrunch;
- IPHost Network Monitor;
- NetMRI;
- NetQoS Performance Center;
- OPNET ACE Live;
- Opsview;
- Performance Co-Pilot;
- Scrutinizer;
- Orion;
- Zenoss.

Таблиця 1.1 – Порівняльний аналіз систем моніторинга

Система моніторингу	Параметри								
	1	2	3	4	5	6	7	8	9
Argus	+	-	-	-	+	+	+	+	+
Intellipool Network Monitor	+	-	-	+	-	+	+	-	+
AdRem NetCrunch	-	+	-	+	-	+	+	+	-
IPHost Network Monitor	+	+	-	+	-	+	-	+	-
NetMRI	-	+	-	+	-	+	+	+	+
NetQoS Performance Center	+	+	+	+	-	+	+	+	+
OPNET ACE Live	+	+	+	+	-	+	-	+	+
Opsview	+	+	-	+	+	+	+	+	+
Performance Co-Pilot	-	+	-	-	+	-	-	+	+
Scrutinizer	+	+	-	-	-	+	+	+	+
Orion	+	+	+	+	+	+	+	+	+
Zenoss	+	+	+	+	-	+	+	+	+

Порівняння систем моніторингу проводилось за наступними параметрами:

- Формування звітів SLA (Service Level Agre-ement) згідно з вимогами. Контроль гарантованих параметрів якості обслуговування SLA, що визначають межоператорские взаємини.
- Формування трендів. Виявлення основних тенденцій динаміки показників якості роботи телекомунікаційної мережі.
- Прогнозування трендів. Прогнозування зміни динаміки показників якості роботи телекомунікаційної мережі.
- Аналіз топології мережі. Збір інформації про елементи мережі.
- Використання агентной моделі моніторингу. Наявність пристроїв, які здійснюють збір і передачу інформації про роботу мережі.

- Підтримка SNMP. Використання протоколу SMNP для обміну інформацією про стани об'єктів спостереження в режимі реального часу.
- Протоколювання подій. Формування докладних записів про стан елементів мережі.
- Датчики позаштатних ситуацій. Наявність пристроїв для оповіщення про виникнення критичних ситуацій, негативної тенденції зміни показників якості роботи телекомунікаційної мережі.
- Розподілений моніторинг. Моніторинг сигнального обміну на предмет відповідності роботи обладнання певним специфікаціям протоколів.

Проведений аналіз показав, що запропоновані на світовому ринку системи моніторингу схожі за виконуваними функціями. Всі вони надають майже однаковий мінімальний набір можливостей, проте кожна з них має певні недоліки: в більшості систем взагалі не реалізовані можливості прогнозування трендів, а в системах, де вони реалізовані, побудова відбувається на основі застарілої статистичної інформації. Подібне прогнозування не враховує фрактальность трафіку, нелінійність характеристик і нестаціонарність процесів.

Узагальнивши пропоновані рішення, можна синтезувати загальну архітектуру системи моніторингу та управління. Всі розглянуті системи моніторингу засновані на використанні агентного підходу. Агенти збирають статистичну інформацію про роботу елементів мережі і передають її в центральну базу даних, після чого вона обробляється керуючими модулями. До складу системи моніторингу повинні входити такі компоненти: формування звітів, модуль управління SNMP, архів і консоль управління. Модуль формування звітів дозволяє формувати з наявних даних інформацію для прийняття управлінських рішень. Модуль управління SNMP відповідає за збір інформації з агентів моніторингу та взаємодія з системами управління.

Архів дозволяє упорядкувати зберігання статистичної інформації та організувати подальшу роботу з нею. Консоль управління реалізує функції конфігурації і управління системою.

У загальному вигляді система моніторингу, яка задовольнить сучасним вимогам, представлена на рисунку 1.1.

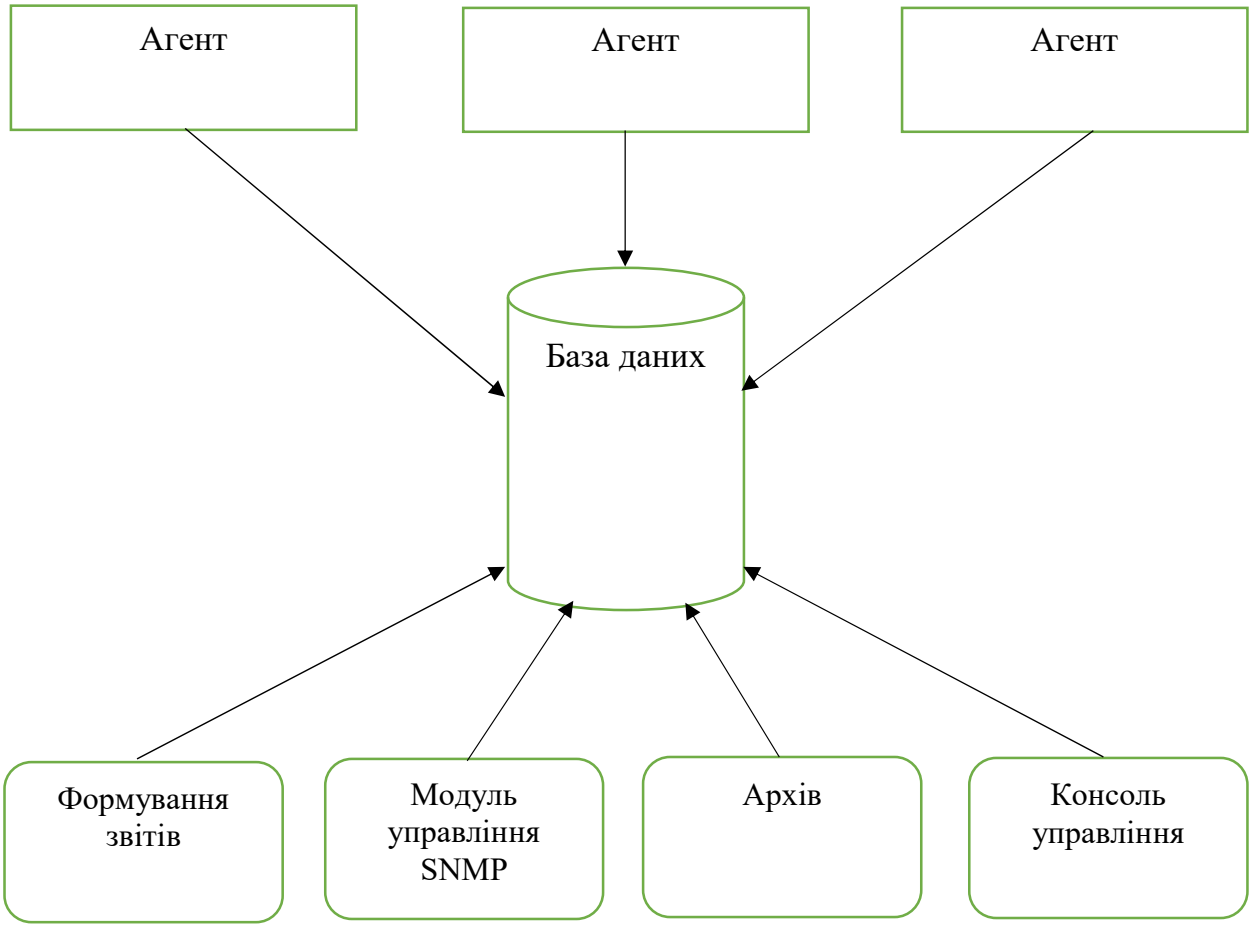

Рисунок 1.1 – Загальна архітектура системи моніторинга

Впровадження систем моніторингу дозволяє вирішити безліч завдань, в числі яких:

- скорочення термінів і витрат на виконання поточних завдань, включаючи активацію послуг;
- підвищення віддачі від існуючих ресурсів мережі і поліпшення якості планування їх майбутнього розвитку;

- зниження потреби в персоналі і, як наслідок, скорочення поточних витрат;
- більш повна реалізація потенціалу сучасного мережевого обладнання за рахунок розробки та реалізації нових послуг;
- зведення до мінімуму ризиків втрат доходів;
- скорочення термінів реагування на що відбуваються в мережі події;
- залучення високоприбуткових клієнтів за рахунок надання додаткових послуг на основі гарантованої якості;
- скорочення термінів введення в експлуатацію нових послуг;
- підвищення якості та оперативності обслуговування користувачів мережі за рахунок чіткої координації та інформаційної підтримки робіт;
- забезпечення координації взаємодії численного персоналу віддалених підрозділів в режимі реального часу.

1.4. Огляд програм моніторингу мережевого трафіку

1.4.1. BMExtreme

Це нова назва добре відомої багатьом програми Bandwidth Monitor. Раніше програма поширювалася безкоштовно, тепер же вона має три версії, і безкоштовної є тільки базова. У цій версії не передбачено ніяких можливостей, окрім, власне, моніторингу трафіку, тому навряд чи можна вважати її конкурентом інших програм. За замовчуванням BMExtreme стежить як за Інтернет-трафіком, так і за трафіком в локальній мережі, проте моніторинг в LAN при бажанні можна відключити. Ця програма представлена на рисунку 1.2.

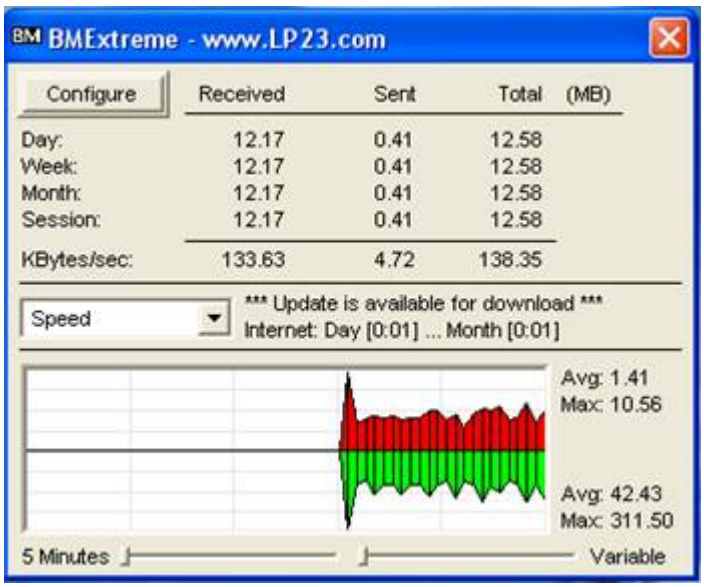

Рисунок 1.2 – Інтерфейс програми BMExtreme

1.4.2. BWMeter

Ця програма має не одне, а два вікна стеження за трафіком: в одному відображається активність в Інтернеті, а в іншому - в локальній мережі. Програма представлена на рисунку 1.3.

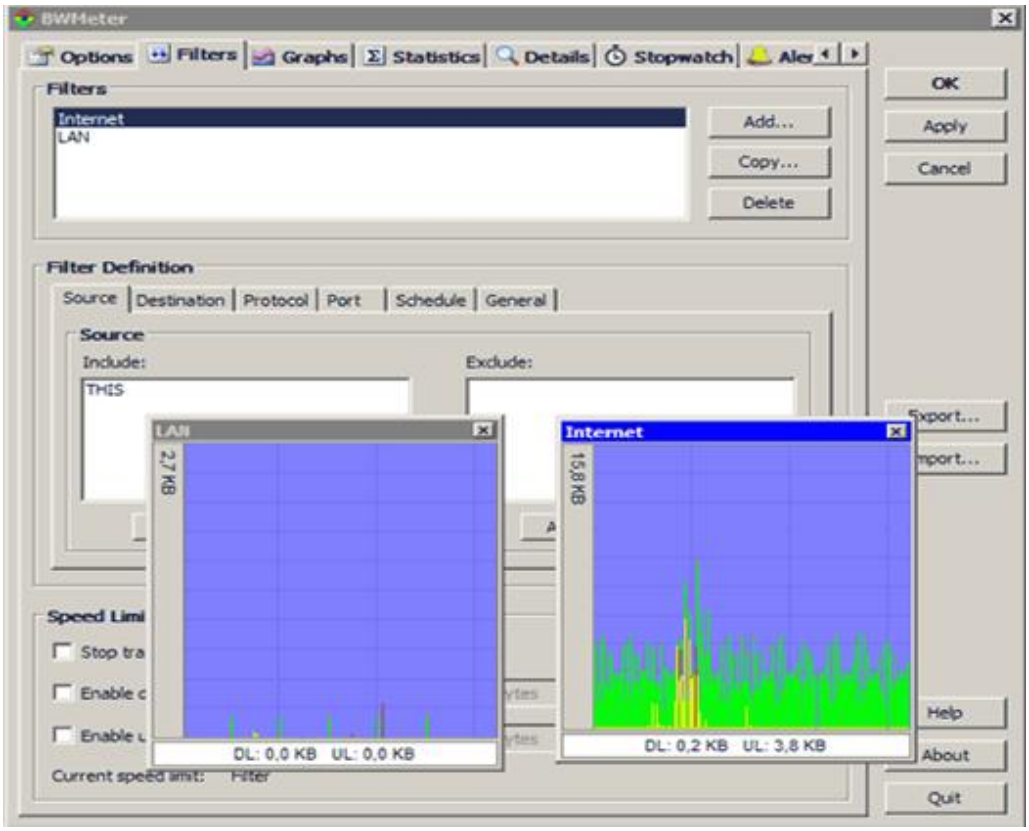

Рисунок 1.3 – Інтерфейс програми BWMeter

Програма має гнучкі налаштування для моніторингу трафіку. З її допомогою можна визначити, чи потрібно стежити за прийомом і передачею даних в Інтернет тільки з цього комп'ютера або зі всіх комп'ютерів, підключених до локальної мережі, встановити діапазон IP-адрес, порти і протоколи, для яких буде чи не буде проводитися моніторинг. Крім цього, можна відключити стеження за трафіком в певні години або дні. Системні адміністратори напевно оцінять можливість розподілу трафіку між комп'ютерами в локальній мережі. Так, для кожного ПК можна задати максимальну швидкість прийому і передачі даних, а також одним клацанням миші заборонити мережеву активність.

При досить мініатюрному розмірі програма має величезний безліччю можливостей, частина з яких можна представити так:

- Моніторинг будь-яких мережевих інтерфейсів і будь-якого мережевого трафіку.
- Потужна система фільтрів, що дозволяє оцінити обсяг будь-якій частині трафіку - аж до конкретного сайту в зазначеному напрямку і пропускної здатності з кожної машини в локальній мережі в зазначений час доби.
- Необмежена кількість параметрів графіків активності мережевих з'єднань на основі вибраних фільтрів.
- Управління (обмеження, призупинення) потоком трафіку на будь-якому з фільтрів.
- Зручна система статистики (від години до року) з функцією експорту.
- Можливість перегляду статистики віддалених комп'ютерів з BWMeter.
- Гнучка система оповіщень і повідомлень по досягненні певної події.

1.4.3. Bandwidth Monitor Pro

Її розробники дуже багато уваги приділили настройці вікна моніторингу трафіку. По-перше, можна визначити, яку саме інформацію програма буде постійно показувати на екрані. Це може бути кількість отриманих і переданих даних (як окремо, так і в сумі) за сьогодні і за будь-який вказаний проміжок часу, середню, поточну і максимальну швидкість з'єднання. Якщо у вас встановлено кілька мережних адаптерів, ви можете стежити за статистикою для кожного з них окремо. При цьому, потрібна інформація для кожної мережевої карти також може відображатися у вікні моніторингу. Програма представлена на рисунку 1.4.

Окремо варто сказати про систему оповіщень, яка реалізована тут дуже вдало. Можна задавати поведінку програми при виконанні заданих умов, якими можуть бути передача певної кількості даних за вказаний період часу, досягнення максимальної швидкості завантаження, зміна швидкості з'єднання та ін. Якщо на комп'ютері працює декілька користувачів, і необхідно стежити за загальним трафіком, програму можна запускати як службу. В цьому випадку Bandwidth Monitor Pro буде збирати статистику всіх користувачів, які заходять в систему під своїми логінами.

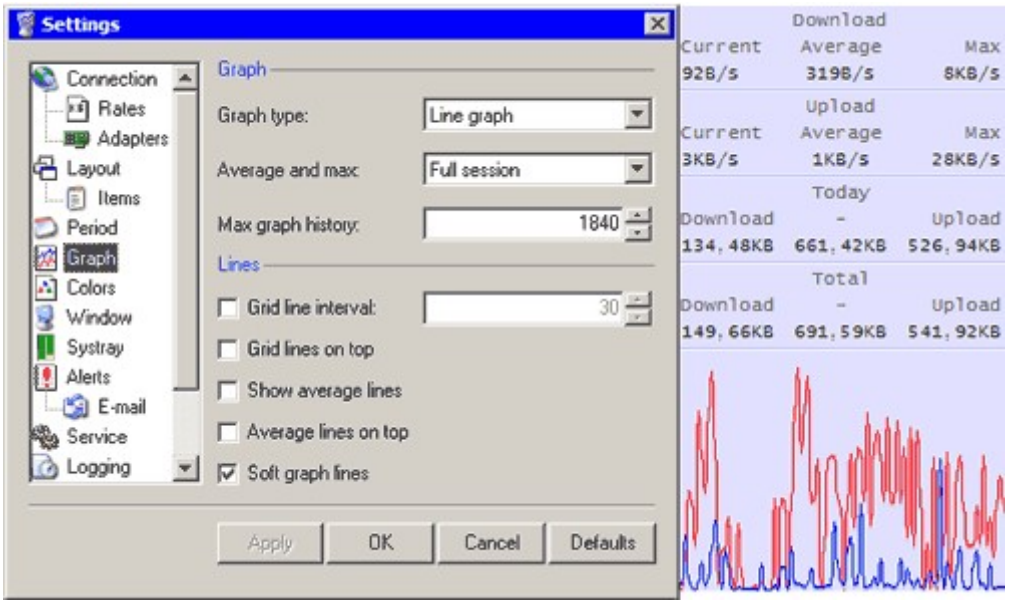

Рисунок 1.4 – Інтерфейс програми Bandwidth Monitor Pro

1.4.4. DUTraffic

Від всіх програм огляду DUTraffic відрізняє безкоштовний статус. Програма представлена на рисунку 1.5.

Як і комерційні аналоги, DUTraffic може виконувати різноманітні дії при виконанні тих чи інших умов. Так, наприклад, він може програвати аудіофайл, показувати повідомлення або ж розривати з'єднання з Інтернетом, коли середня або поточна швидкість завантаження менше заданого значення, коли тривалість Інтернет-сесії перевищує вказане число годин, коли передано певну кількість даних. Крім цього, різні дії можуть виконуватися циклічно, наприклад, кожен раз, коли програма фіксує передачу заданого обсягу інформації. Статистика в DUTraffic ведеться окремо для кожного користувача і для кожного з'єднання з Інтернетом. Програма показує як загальну статистику за вибраний проміжок часу, так і інформацію про швидкість, кількість надісланих і отриманих даних і фінансових витратах за кожну сесію.

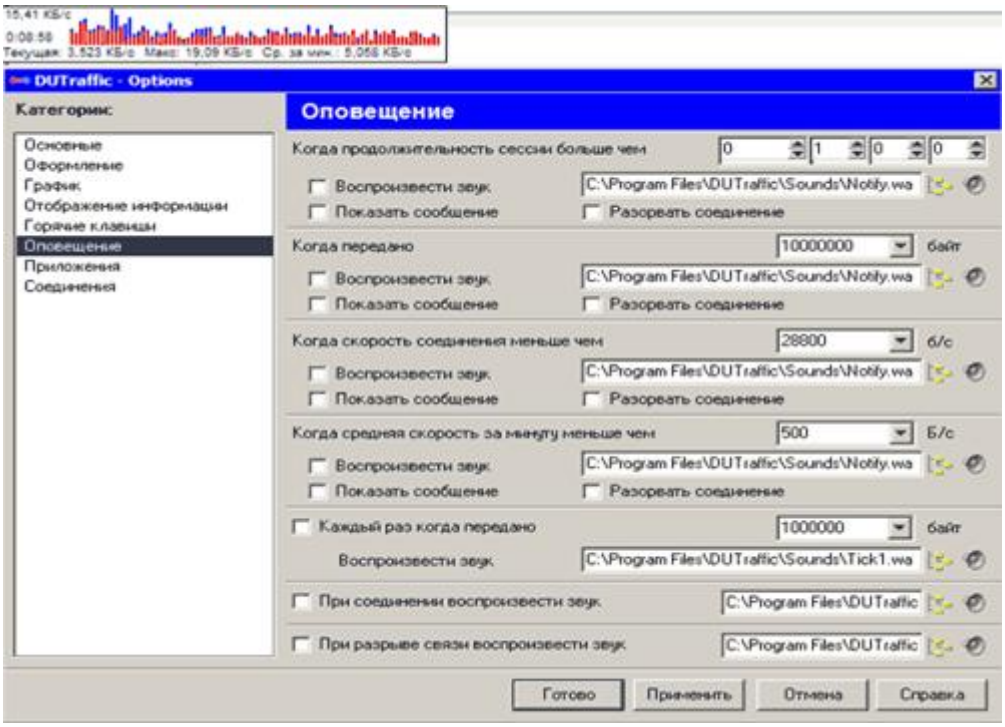

Рисунок 1.5 - Інтерфейс програми DUTraffic

1.4.5. Система моніторингу Cacti

Cacti це open-source веб-додаток (відповідно відсутній інсталяційний файл). Cacti збирає статистичні дані за певні часові інтервали і дозволяє відобразити їх у графічному вигляді. Система дозволяє будувати графіки за допомогою RRDtool. Переважно використовуються стандартні шаблони для відображення статистики по завантаженню процесора, виділенню оперативної пам'яті, кількістю запущених процесів, використання вхідного / вихідного трафіку.

Інтерфейс відображення статистики, зібраної з мережевих пристроїв, представлений у вигляді дерева, структура якого задається самим користувачем. Як правило, графіки групують за певними критеріями, причому один і той же графік може бути присутнім в різних гілках дерева (наприклад, трафік через мережевий інтерфейс сервера - в тій, яка присвячена загальній картині інтернет-трафіку компанії, і в галузі з параметрами даного пристрою) . Є варіант перегляду заздалегідь складеного набору графіків, і є режим попереднього перегляду. Кожен з графіків можна розглянути окремо, при цьому він буде представлений за останні день, тиждень, місяць і рік. Є можливість самостійного вибору часового проміжку, за який буде згенеровано графік, причому зробити це можна, як вказавши календарні параметри, так і просто виділивши мишкою певну ділянку на ньому.

В цілому можна сказати, що більшості домашніх користувачів буде достатньо можливостей, які надає Bandwidth Monitor Pro. Якщо ж говорити про саму функціональної програмою для моніторингу мережевого трафіку, це, безумовно, BWMeter.

З числа розглянутих програм-аналізаторів мережевого трафіку хотілося б виділити Wireshark, яка має більшу кількість функціональних можливостей.

Система моніторингу Cacti максимально відповідає підвищеним вимогам, які пред'являються в разі проведення дослідження мережевого трафіку в наукових цілях.

2. СТРУКТУРА ТА АЛГОРИТМИ СТВОРЕННЯ ЗАСОБУ МОНІТОРИНГУ

Засіб моніторингу, який був створений в даній роботі, представлений у вигляді веб-орієнтованої системи з інтерактивною картою. Він складається з трьох частин: з клієнтської частини, з серверної частини та Бази даних або системи управління базами даних, сокр. СУБД (англ. Database Management System).

2.1. Структура та алгоритми створення клієнтської частини.

Клієнтська частина веб-орієнтованої системи являє собою графічний інтерфейс, який відображається користувачу на веб – сторінці за допомогою спеціального додатку – веб-браузеру. Користувач взаємодіє з веб-системою через браузер, коли зробить якусь дію, наприклад, натисне кнопку або перейде по якомусь посиланню.

Клієнтська частина створеної веб-орієнтованої системи складається з верстки веб-сторінок та адаптивної верстки (написання HTML і CSS коду та написання медіа-запросів), програмування за допомогою JavaScript та бібліотеки JQuery.

Верстка веб-сторінок – це створення повноцінної html-сторінки на основі розробленого дизайну в графічному редакторі.

Для того, щоб зверстати веб-сторінку, треба використати графічний редактор Photoshop, мову розмітки гіпертексту HTML та каскадні таблиці стилів CSS.

Спочатку веб-сторінки являли собою просто набір тегів. Елементи відображалися в тому порядку, в якому вони записувалися в HTML коді. Візуально, інформацію поділяли за допомогою горизонтальних ліній або відступів.

Другим етапом побудови сайтів стала побудова на основі таблиці. Сайт був таблицю в осередки якої вставлялася інформація. Це дозволяло реалізовувати колонки і окремі блоки, проте зручність роботи з верстkою на

основі таблиці досить низька, не є гнучким. Також, таблична верстка не придатна для побудови адаптивних сайтів. В даний час така верстка вважається застарілою.

Третій, найбільш сучасний підхід, до верстки сайтів полягає в побудові сайту за допомогою блоків, або блокової верстці. В англомовних джерелах такий підхід називають Layouts. Блоки при такому підході розташовуються один під іншим, або змінюють їх порядок відображення за допомогою позиціонування в CSS

Блокова верстка - це підхід, при якому сайт будують на основі блоків, як блоки виступають теги div. Теги div мають деякі властивості:

- Блок div - блоковий елемент, тому, якщо ширина не задається, то розтягується на повну ширину вікна браузера;
- Блок div - висота блоку, коли її не вказано, дорівнює вмісту. Порожній блок div має висоту - 0 px, тому не відображається на сторінці;
- Блок div не має оформлення. Щоб його побачити потрібно задати йому стилі в CSS.
- Блок div не несе смислового навантаження, це просто спосіб структурувати сайт
- Блок div - може містити будь-яку кількість вкладених елементів. Так в div можна вкладати інші блоки div, заголовки, параграфи, зображення і багато інших елементів

Верстка вважається якісною, якщо виконуються такі пункти:

- Верстка створюється із застосуванням тегів .
- Верстка адаптивна та кросбраузерна - сторінка поводиться коректно в різних браузерах і на мобільних гаджетах.
- Верстка валідна – без помилок в коді.
- Текстова. Те, що можна зверстати у вигляді тексту, що не верстається картинкою. Пошукова система любить текстовий

контент, тому даний факт повинен враховувати кожен верстальник.

- Код мінімально короткий, всі стилі винесені в окремий файл.
- Весь вміст в html і css прописано малими літерами.
- Для тега <img> (картинки) обов'язково вказані висота і ширина зображення.
- CSS використовується переважно. Тобто, якщо можна обійтися без JS - то динаміка описується в CSS.
- Те ж саме для картинок. Якщо спецефекти для шрифту можна накласти за допомогою CSS, то перевага надається тексту, а не картинка.
- JS файли підключені в низу коду. Якщо ви підключаєте їх в <head>, то це погано позначається на швидкості завантаження веб-сторінки.
- JS файли об'єднані в один (по можливості).
- Навігація по сайту реалізована списками (<ul>, <li>), пошукова система швидше розуміє таку навігацію.
- Правильна робота з заголовками. Щоб у SEO-фахівця в майбутньому не було проблем, то треба вказувати H1-H6 тільки в тематичній частині сайту. Розміщуйте заголовки за принципом: першим йде H1, потім по спадаючій інші види заголовків.
- Продумано стилі всіх заголовків, абзаців, картинок, списків в тематичній частині документа.
- Структурований код, який наочно показує всі закриваючі і відкриваючі теги.

Зараз при написанні HTML коду вже сміливо можна використовувати теги й елементи розмітки, які з'явилися разом з стандартом HTML5. На етапі написання HTML ми, як би, створюємо скелет сторінки, її абстрактну модель за допомогою тегів (мови розмітки HTML). При написанні розмітки ми так само відразу можемо прописувати елементам класи і ідентифікатори.

У всьому проекті має бути порядок: від структури проекту до імен класів, маркерів та написання коду. Якщо під час вибору важливо стежити за типом інформації та помістити його у відповідні блоки (заголовок, список, посилання, рядок-позиція, абзац і тощо). Важливо зберігати здоровий глузд при класах і ідентифікаторів назв. Класи повинні дати абстрактне поняття блоку, до якого вони належать, так що код легше читати, а потім писати стилі. В принципі, не повинно бути нічого складного, якщо ми виберемо меню, то логічно, що містячий блок дає клас .nav або .navigation, якщо це блок з текстом, то ми можемо надати йому клас .block-text і тощо.

На сьогоднішній день є один популярний підхід, який стосується принципів побудови проекту в цілому, але на даному етапі нас цікавить саме іменування класів. Підхід називається БЕМ і розшифровується, як Блок Елемент Модифікатор.

БЕМ надає єдині правила створення і зберігання коду, які допомагають масштабувати і повторно використовувати код, збільшити продуктивність і спростити командну роботу. Навіть якщо вся ваша команда - це ви самі, БЕМ може бути вам корисний.

БЕМ допомагає вирішити такі завдання:

- повторно використовувати верстку;
- безболісно міняти верстку місцями в одному проекті;
- переносити готову верстку з проекту в проект;
- створювати стабільний, передбачуваний і зрозумілий код.

У БЕМ-проект будь-якої інтерфейс ділиться на блоки, які можуть містити елементи. Блоки - це незалежні компоненти сторінки. Елементи не існують поза блоком. Кожен елемент може належати тільки одному блоку.

Саме блокам і елементам присвячені перші дві букви в абревіатурі БЕМ.

Ім'я блоку завжди унікально. Воно задає простір імен для елементів та встановлює видиму зв'язок між усіма складовими блоку. Довгі, але зрозумілі

імена блоків і елементів дозволяють визначити зв'язок між компонентами і не втратити складові частини цих компонентів при перенесенні верстки.

Правила іменування класів підводять нас до наступного етапу. Коли написана html структура проекту, визначені класи можна переходити до написання CSS стилів і нарізці макета.

Варто згадати про 2-х CSS файлах-доповненнях:

- Reset.css. Метою цього набору правил є скидання стилю браузера, в якому елементи тегу відображаються за умовчанням. Отже, коли ви використовуєте reset.css, вам не потрібно переписати стилі браузера, в основному ми працюємо над "чистим аркушем", і ми можемо сконцентруватися на написанні власних стилів з нуля.
- Normalize.css. Цей файл не тільки задає стилі за замовчуванням там де це необхідно, а й виправляє деякі недоліки старих браузерів.

Ці два файли не зовсім коректно порівнювати, так як у них різна "філософія".

Відмінності normalize.css від reset.css:

- На відміну від reset.css, після підключення normalize.css, ви візуально зможете визначити де у вас знаходяться різні елементи, ті ж параграфи мають зовнішні відступи. Тому, застосовуючи «таблицю стилів зі скиданням», ви просто приведете велику кількість елементів до одного виду.
- Виходячи з 1 правила випливає, що нам потрібно написати стилі з нуля, тому що у нас все скинуто, що особисто на мене, ця дія займає багато часу.
- У normalize.css є виправлення різних загальних помилок, таких як відображення HTML5 тегів тих же форм
- Коли ви працюєте з інструментами налагодження в браузерах, підключаючи reset.css ви отримаєте величезне «простирадло» з

властивостей, яке в підсумку збільшить ваш час на написання нових стилів і пошук старих.

– У normalize.css кожне окреме правило задокументовано і ви легко зможете зрозуміти для чого воно там, і якщо ви впевнені що воно вам не потрібно, то зможете легко його видалити.

У обох файлів з правилами є свої плюси і мінуси, на даний момент популярний normalize.css. Популярність даного зводу правил обумовлена ще й тим, що вам не доводиться при розробці заново прописувати основні властивості для базових елементів сторінок, а лише модифікувати їх за потребою.

Так само варто відзначити, що всі розміри і відступи беруться безпосередньо з дизайн-макету. Для отримання даних значень нам необхідно використовувати інструмент «лінійка» і напрямні (мова про інструментарій Adobe Photoshop), а потім переносити отримані значення в код. Якщо ми працюємо з фіксованим макетом, то значення переносяться в пікселях як є, якщо ж у нас «гумовий», то значення потрібно переводити в відсотки.

В даний час відвідувачі заходять на сайти не тільки з настільного комп'ютера, але і з ноутбука, планшета, смартфона. При цьому на пристроях з маленьким екраном часто текст занадто дрібний, посилання практично неклікабельни, елементи управління розташовані дуже близько один від одного.

Щоб усунути ці недоліки стали розробляти окремі мобільні версії сайтів. Але це довго, дорого і незручно в підтримці.

Рішення виниклої проблеми - адаптивна верстка, при якій CSS-стилі змінюються динамічно для різної ширини вікна браузера.

Розробку адаптивної верстки найчастіше проводять від вже існуючого сайту для великого екрану до екранів меншого розміру. Перевіряти результат роботи можна різними способами:

– просто міняти розмір екрана браузера;

- розташувати праворуч панель інспектора компонентів і змінювати її ширину;
- використовувати адаптивний дизайн браузера, натиснувши Ctrl + Shift + M в Firefox або Google Chrome.

Мобільні браузери, типові для сторінку сайту за сторінку для звичайного комп'ютера і масштабує її по ширині екрану телефону. В результаті текст стає дрібним, і відвідувачеві, щоб його прочитати, доводиться збільшувати масштаб сторінки.

Для коригування розмірів і масштабування сторінки з урахуванням ширини екрану пристрою використовують метатег viewport, який містить інструкції для браузера.

Щоб повідомити браузеру, що сторінка адаптується до будь-яких пристроїв, в заголовок документа додають метатег viewport.

<meta name=”viewport” content=”width=device-width, initial-scale=1.0”>

Таблиця 2.1 - Основні властивості метатега viewport

width	Ширина видимої області. Рекомендоване значення: device-width.
height	Висота видимої області. Рекомендоване значення: device-height.
initial-scale	Початковий масштаб сторінки. Приймає значення від 1 до 5. Рекомендоване значення: 1.
User-scalable	Дозволяє або забороняє можливість масштабування сторінки. Приймає значення true або false.
Maximum-scale	Максимальний масштаб сторінки. Приймає значення, яке має бути більше або рівним initial-scale. Значення 1 забороняє збільшення масштабу сторінки.

Можливість застосовувати різне оформлення залежно від ширини вікна дає CSS-стандарт Media Queries.

Медіа-запит починається з правила @media, після якого слід умова застосування стилів, що складається з типу носія (в наведеному прикладі all), логічного оператора (and) і медіа-функції (max-width: 360px).

Таблиця 2.2 – Логічні оператори media queries

And	Вказується для об'єднання декількох умов.
Not	Вказується для заперечення умови. Оператор not оцінюється в запиті останнім.
Only	Ключове слово для старих браузерів, які не підтримують медіа-запити і вважають only типом носія. Але так як такого типу не існує, то весь стиль цілком ігнорується. Сучасні браузери сприймають медіа-запит з only і без only однаково.

Таблиця 2.3 – Основні медіа-функції

Width	Ширини окна браузера
Device-width	Ширина екрана девайсу
Height	Висота окна браузера
Device-height	Висота екрана девайсу

Кінцевим кроком є написання скриптів JS. Під час створення веб-сайтів стало практично стандартним використання бібліотеки jQuery, що дозволяє легко маніпулювати елементами веб-сайту (вузлами DOM), призупиняти прослуховування подій, надсилати запити на сервер, обробляти результати роботи тощо. Але ви не повинні сліпо довіряти тенденціям. Сьогодні нативна підтримка JavaScript досягла такого рівня, що jQuery більше не може бути

потрібним. Тому перед тим, як сліпо прив'язати jQuery, слід подумати про те, чи це необхідно для ваших завдань та чи недостатньо вбудованих функцій JavaScript.

Важливо пам'ятати, що не рекомендується використовувати JavaScript для стилю, тобто для DOM (елемент сторінки) не потрібно додавати портянку властивостей CSS за допомогою JavaScript, щоб виділити його статус тощо. Рекомендується використовувати класи, це заздалегідь в CSS для визначення класу стану (активний, неактивний, прихований, використаний і т. д.), а при маніпуляції елементами просто додавайте або видаляйте відповідні класи.

Знову ж таки, повернувшись до першої теми Mobile, не можна не згадати двох концепцій, котрі пов'язані з цією технікою. Прогресивне зміцнення і витончена деградація, що означає поступове поліпшення і постійне погіршення стану. Ці правила описують два різних підходи до розвитку: у першому випадку ми створюємо наш веб-сайт, створюємо сценарії, що містять старі веб-переглядачі та системи, або визначаємо поведінку, коли неможливо запустити скрипти, а потім поступово покращувати сценарій, використовуючи сучасні функції та методи, тому ми отримуємо веб-сайт що буде працювати однаково добре в старих і нових браузерах та середовищах (те ж саме стосується і css).

Після написання css, javascript та html коду до нашої веб-сторінки треба перевірити правильність написання та чи все зроблено правильно. Для цього можна використати online засоби:

- Для перевірки JS: http://www.jslint.com/
- Для перевірки html: https://validator.w3.org/
- Для перевірки CSS: http://jigsaw.w3.org/css-validator/

Завдяки цим службам ви можете перевірити, чи не забули ви закрити маркер де-небудь, незалежно від того, чи правильно ви використовуєте параметри та атрибути, якщо все впорядку з вашими стилями та правилами, а також перевірити код правильності написання функцій, методів тощо.

2.2. Технології для створення клієнтської частини

Для побудови клієнтської частини веб-орієнтованої системи використовується HTML-розмітка, CSS-стилі, JavaScript, JQuery, Ajax, LESS, Bootstrap та Adobe Photoshop.

2.2.1. HTML

HTML (англ. HyperText Markup Language, тобто мова розмітки гіпертексту) – це проста мова програмування , основна задача якої є форматування сторінки при відображенні в веб-браузері, наприклал у Google Chrome. HTML дозволяє відобразити інформаційний контент – текстові блоки, посилання і зображення. Написання веб-сторінок на HTML не вимагає інтерпретації вихідного коду в двійковий код. Мова розмітки гіпертексту за визначенням повинна інтерпретуватися браузером. Це, безумовно, накладає деякі обмеження на можливості мови і на сумісність нових конструкцій зі старими версіями браузерів. Сучасні веб-сторінки вже не обходяться одним тільки HTML. Його гармонійно доповнюють засоби динамічного HTML: скрипт мови JavaScript, каскадні таблиці стилів (CSS), іноді присутні Java-аплети.

HTML складається з елементів. Елементи складаються, в свою чергу, з тегів, атрибутів і вмісту (останнє не завжди обов'язково).

Теги - це прості дескриптори, які вказують веб-браузеру на розташування елементів та їх форматування.

- Відкриваючий тег. Початок кожного елемента в HTML позначається тегом, який відкривається тегом: ім'я елемента вказується в кутових дужках. Наприклад, відкриваючий тег абзаца – це </p>, а відкриваючий тег зображення - <img>.
- Закриваючий тег. Кінець елемента в HTML позначається тегом, який закривається. Він точно такий же, як відкриваючий, тільки перед ім'ям елемента ставиться коса риска. Отже, </p> - тег, що закриває абзац. У деяких елементів, наприклад у тега IMG, закриваючих тегів немає.

- Атрибути. За допомогою атрибутів визначають параметри елементів. Додаючи відповідні атрибути до тегу, задають висоту і ширину зображень, як, наприклад в наступному фрагменті програмного коду: <IMG width=”100” height=”100”>
- Вміст. Вміст розташований між закриваючим і відкриваючим тегами, наприклад, в наступному фрагменті програмного коду: Вміст тегу

2.2.2.CSS та LESS

Як буде виглядати шрифт тексту, як буде розташовуватися сам текст, де будуть знаходитися різні об'єкти, - за все це відповідає CSS (Cascading Stylesheets). Він розроблений для того, щоб розширити можливості по оформленню веб-сторінок.

З появою каскадних таблиць стилів (CSS) все змінилося. Тепер веб-майстри можуть оформляти сторінку так, як вважають за потрібне. Ці нововведення дають більше свободи розробникам, проте при цьому привносять і деяке безладдя.

Існує три способи додавання каскадних стилів на веб-сторінку:

- Внутрішні стилі. Стиль застосовується тільки до того тегу, в якому він визначений, і абсолютно не впливає на інші елементи. Це найпростіший варіант реалізації, але він не забезпечує достатньої гнучкості. Приклад: <h1 style=”font-size: 20px;”></h1>;
- Оголошення стилю за допомогою тега <style>. В даному випадку форматування нівелюється при наявності величезного сайту з безліччю веб-сторінок. Тег розміщується в HTML-документі всередині тега <head>. Оголошення стилю буде виглядати приблизно так: <head><style> h1{color: blue} </style></head>;
- Окремий файл. Цей файл необхідно пов'язати з файлами .html веб-сайту. Даний підхід припускає створення .css-файлу, в якому визначені всі стілі. Це дає величезні переваги в порівнянні з двома іншими підходами, описаними раніше: один .css-файл управляє

відображенням безлічі веб-сторінок. Залишається лише зробити посилання на .css-файл на кожній сторінці. Після цього всі зміни, що вносяться до нього, автоматично призведуть до зміни форматування пов'язаних веб-сторінок.

При використанні HTML і CSS важливо розуміти, наступне:

- HTML-код формує текст логічно, тобто задає структури Web-сторінки: розташування і порядок проходження абзаців, графічних зображень, рядків і осередків в таблиці і особливе значення окремих фрагментів тексту.
- Таблиці стилів CSS формують тексти фізично, тобто задають уявлення Web-сторінки: яким шрифтом будуть набрані звичайний текст абзаців, яким кольором виділити заголовки, чи будуть у таблиці рамка тощо.
- Каскадні таблиці стилів по суті своїй нединамічні. Вони дозволяють визначати, як буде виглядати документ при завантаженні і не більше того. Але властивості веб-сторінок, створених за допомогою CSS, можна динамічно змінювати за допомогою мови JavaScript.

LESS – це інструмент, що перетворює код з одного синтаксису в інший. Зазвичай, на вхід препроцесора надходить код, написаний з використанням синтаксичних конструкцій, зрозумілих цього препроцесору.

Препроцесор може повністю заміщати синтаксичні конструкції мови або частково розширювати їх, тобто додавати нові конструкції. Так як препроцесор перетворює код, то крім входу у нього повинен бути і вихід. Інакше який сенс в його роботі, адже інші програми не зможуть скористатися його працями.

Як правило, на виході очікується код з більш низьким рівнем. Тобто код, позбавлений синтаксичних конструкцій, що вносяться препроцесором.

LESS додає багато потрібних динамічних властивостей в CSS. Він вводить змінні, операції, function-like елементи і домішки. Можливість писати таблиці стилів модульно позбавить вас від клопоту.

Є два способи використання LESS. Ви можете створити LESS файл і конвертувати його за допомогою Javascript на льоту або скомпілювати його заздалегідь і використовувати отриманий CSS файл.

Для використання LESS та JavaScript файл потрібно завантажити з сайту LESS Javascript файл і прив'язати його до сторінки як будь-який інший js скрипт.

<script src=”less.js” type=”text/javascript”></sript>

Потім створимо файл з розширенням .less і прив'яжемо його за допомогою такого коду:

<link rel=”stylesheet/less” type=”text/css” href=”style.less”>

Тепер LESS файл буде працювати також як і звичайний CSS.

2.2.3.JavaScript та JQuery

JavaScript (англ. JS) - це повноцінна динамічна мова програмування, яка застосовується до HTML документу, і може забезпечити динамічну інтерактивність на веб-сайтах. Її розробив Brendan Eich, співзасновник проекту Mozilla, Mozilla Foundation і Mozilla Corporation.

JavaScript має величезні переваги в порівнянні з іншими мовами програмування, які дозволяють додавати інтерактивні компоненти на веб-сторінку, оскільки спочатку розроблявся саме для цих цілей. Обробка здійснюється в веб-браузері користувача, тому немає ніякого навантаження на сервер.

JavaScript, в свою чергу, потрібен для того, щоб сайт міг взаємодіяти з користувачем - аналізувати введені дані, перевіряти їх коректність, змінювати контент сторінки відповідно до запитів, видавати оповіщення, модальні вікна і тощо.

При роботі з JavaScript в код веб-сторінки потрібно додати два компоненти: сам скрипт і процедуру, яка буде його запускати.

Тіло скрипта розміщується в контейнерному тезі <head>, як у наведеному прикладі нижче, де визначається деяка функція:

<head><script> function what(){}</script></head>

На сторінку повинна бути додана якась процедура, яка запустить скрипт. Вона знаходиться всередині тега <body>. Запуск скрипта може здійснюватися різними способами. Наприклад, активізація скрипта відбувається при зміні елемента інтерфейсу, коли користувач наводить на нього курсор миші.

Популярність мови JavaScript пов'язана з його широкими можливостями по взаємодії з елементами веб-сторінки без її перезавантаження. Це дозволяє ховати і показувати фрагменти дизайну, переміщати їх і міняти оформлення. Шляхом таких дій можна створювати презентаційні ефекти, меню, невеликі гри, обробляти дані форм і керувати вмістом.

JavaScript підтримує повноцінну роботу з cookies - невеликі текстові файли на локальному комп'ютері, в яких зберігається технічна інформація. Cookies можна використовувати для збереження дати останнього відвідування читача, паролів, а також будь-якої інформації про дії користувача на сайті. Подібне застосування дозволяє персоналізувати сайт і зробити його більш зручним для відвідувачів.

JavaScript містить всі необхідні арифметичні операції, підтримує всі стандартні математичні функції, як з цілими числами, так і з плаваючою крапкою.

Переваги JavaScript:

- Жоден сучасний браузер не обходиться без підтримки JavaScript.
- З використанням написаних на JavaScript плагінів і скриптів впорається навіть не фахівець.

- Постійно удосконалюється мова - зараз розробляється бета-варіація проекту, JavaScript2.
- Взаємодія з додатком може здійснюється навіть через текстові редактори - Microsoft Office і Open Office.

Недоліки JavaScript:

- Знижений рівень безпеки через повсюдного і вільного доступу до вихідного коду популярних скриптів.
- Безліч дрібних дратівливих помилок на кожному етапі роботи. Велика частина з них легко виправляється, але їх наявність дозволяє вважати цю мову менш професійною, порівняно з іншими.
- Своєрідним недоліком можна вважати той факт, що частина активно використовуваних програм (особливо додатків) перестануть існувати при відсутності мови, оскільки цілком базуються на ньому.

Для спрощення розробки клієнтської частини сайту можна використовувати бібліотеки, що містять готові збірники компонентів для написання типових програм. Наприклад, безкоштовна бібліотека JQuery містить корисні функції для вирішення типових завдань на JavaScript.

jQuery - це швидка, невелика та багатофункціональна бібліотека JavaScript, що входить до одного файлу .js.

jQuery робить життя веб-розробника легким. Він надає безліч вбудованих функцій, за допомогою яких можна швидко і легко виконувати різні завдання.

Важливі функції JQuery:

- Вибір DOM: jQuery забезпечує Selectors для отримання елемента DOM на основі різних критеріїв, таких як ім'я тесту, ідентифікатор, ім'я класу css, ім'я атрибута, значення, n-й дитини в ієрархії тощо.
- Маніпуляція DOM: Можна керувати елементами DOM за допомогою різних вбудованих функцій jQuery. Наприклад,

додавання або видалення елементів, зміна HTML-контенту, класу CSS і т.д.

- Спеціальні ефекти. Ви можете застосувати спеціальні ефекти до елементів DOM, таких як покази або приховування елементів, зменшення видимості, слайд-ефект, анімація тощо.
- Події: бібліотека jQuery містить функції, які еквівалентні подіям DOM, таких як dblclick, mouseenter, mouseleave, blur, keyup, keydown тощо. Ці функції автоматично обробляють кросбраузерні проблеми.
- Ajax: jQuery також включає в себе прості у використанні функції AJAX для завантаження даних з серверів без перезавантаження всієї сторінки.
- Підтримка між браузерами: бібліотека jQuery автоматично обробляє крос-браузерні проблеми, тому користувачеві не потрібно турбуватися про це. jQuery підтримує IE 6.0+, FF 2.0+, Safari 3.0+, Chrome і Opera 9.0+.

Переваги JQuery:

- Легко навчитися: jQuery легко вивчити, оскільки підтримує той же код JavaScript стилю.
- jQuery надає насичений набір функцій, які підвищують продуктивність розробників, пишучи менш читаний код.
- Відмінна документація API: jQuery забезпечує відмінну документацію в Інтернеті API.
- Ненав'язливий: jQuery є ненав'язливим, що дозволяє відокремити проблеми, розділяючи html та jQuery-код.

2.2.4.AJAX

AJAX (абревіатура від Asynchronous JavaScript and XML) - це технологія взаємодії з сервером без перезавантаження сторінки. Оскільки не

потрібно кожен раз оновлювати сторінку цілком, підвищується швидкість роботи з сайтом і зручність його використання.

Зрозуміти основний принцип роботи AJAX допомагає представлене нижче зображення (рис. 1.2).

В роботі технології можна виділити 4 основних етапи:

- Користувач викликає AJAX. Зазвичай це реалізується за допомогою будь-якої кнопки, що пропонує отримати більше інформації.
- Система відправляє на сервер запит і всілякі дані. Наприклад, може знадобитися завантаження певного файлу або конкретних відомостей з бази даних.
- Сервер отримує відповідь від бази даних і відправляє інформацію в браузер.
- JavaScript отримує відповідь, розшифровує його і виводить користувачеві.

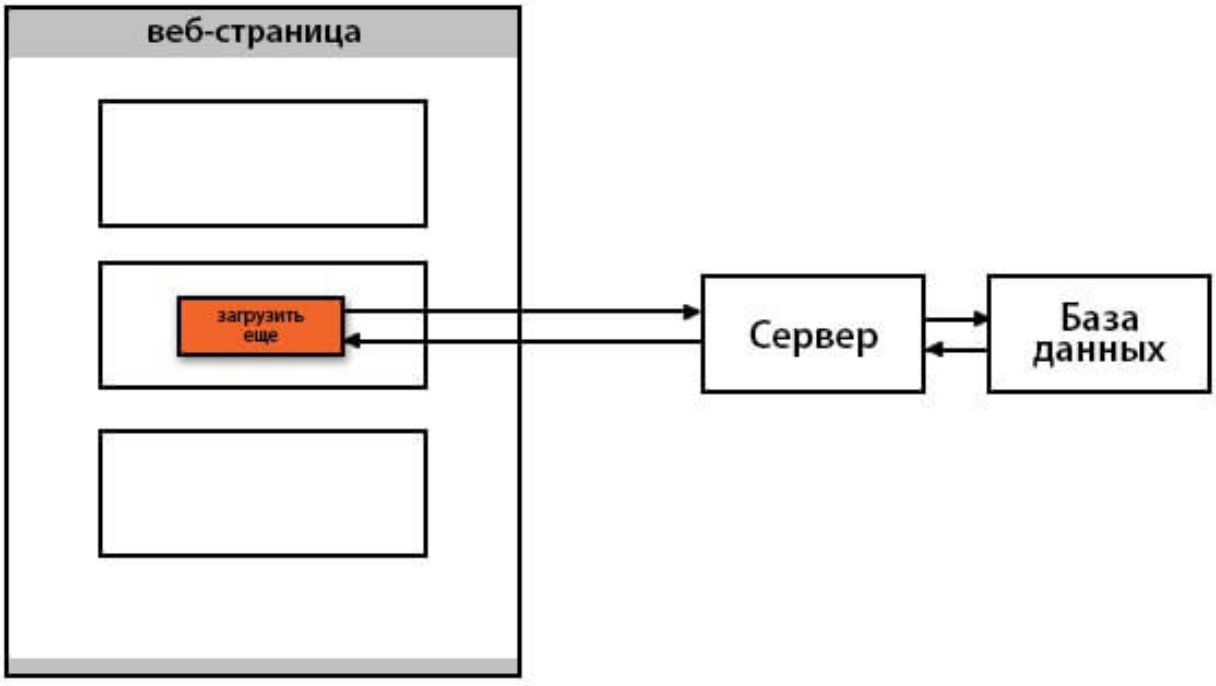

Рисунок 2.1 - Основний принцип роботи AJAX

Для обміну даними на сторінці створюється об'єкт XMLHttpRequest, він буде виконувати функцію посередника між браузером і сервером. Запити можуть відправлятися в одному двох типів - GET і POST. У першому випадку звернення проводиться до документа на сервері, в ролі аргументу йому передається URL сайту. Для запобігання переривання запиту можна скористатися функцією JavaScript Escape. Для великих обсягів даних застосовується функція POST.

Серверна частина обробляє дані, що надходять і на їх підставі створює нову інформацію, яка буде відправлена клієнтові.

AJAX застосовує асинхронну передачу даних. Такий підхід дозволяє користувачеві здійснювати різні дії під час «фонового» обміну інформацією з сервером. Діє оповіщення користувача про протікають процеси, щоб він не подумав, що сайт «завис» або на ньому стався якийсь збій.

Як відповідь сервер використовує простий текст, XML і JSON. У першому випадку результат можна відразу ж відобразити на сторінці. При отриманні XML-документа його зазвичай конвертують в HTML і виводять на екран. Якщо відповідь отримана в форматі JSON, клієнту слід виконати отриманий код. Після цього буде сформовано об'єкт JavaScript.

Переваги технології AJAX:

- Скорочення трафіку. Обсяг даних при роботі з web-додатками значно знижується. Це відбувається за рахунок того, що не потрібно завантажувати всю сторінку цілком, досить отримати тільки змінену частину або набір даних. Після цього JavaScript змінює вміст сторінки в браузері.
- Зниження навантаження на сервер. Грамотне використання AJAX дозволяє багаторазово зменшити навантаження на сервер. Наприклад, можна використовувати шаблон для створення постійних елементів сайту: логотипу, меню і т. П. Крім того, для задоволення конкретного запиту не потрібно оновлювати всю

сторінку. Наприклад, при голосуванні на сайті користувач вибирає потрібний пункт, натискає кнопку, інформація відправляється на сервер, після чого приходить відповідь. Протягом усього цього часу поновлення сторінки не відбувається.

- Збільшення швидкості роботи сервісу. Оскільки подгружается тільки змістовна частина, користувач набагато швидше бачить результат своїх дій.
- Широкий спектр можливостей. Використання AJAX не обмежується формами. Наприклад, при проходженні реєстрації на деяких сервісах користувач вводить логін - і через мить йому видається інформація про те, вільний він чи ні. Також при введенні пошукового запиту в Google після кожної букви або слова пропонується кілька варіантів запиту. Це значно підвищує комфорт роботи з сайтами.

Недоліки AJAX:

- JavaScript повинен бути включений. Сторінки сайтів, створені за технологією AJAX, не можуть нормально працювати при відключеній підтримці JavaScript. На них не можна розмістити закладки, та й пошукові системи далеко не завжди можуть їх проіндексувати.
- Неможливість інтеграції з інструментами браузера. При динамічному формуванні сторінок браузер не відображає їх в історії відвідування, тому кнопка «Назад» не допоможе переміститися на попередній етап роботи. Втім, така проблема може бути вирішена за рахунок спеціальних скриптів. Також відсутня можливість встановити на потрібний матеріал закладку.
- Погіршення безпеки. Помітним недоліком AJAX також є прогалини в безпеці, адже вихідний код кожен може прочитати в браузері.

- Неможливість встановити число звернень. Механізм динамічної підвантаження контенту помітно спотворює дані статистики. Адже в цьому випадку при переміщенні користувача з різних сторінок не виконується операція їх перезавантаження. І звичайний лічильник не реєструє перехід. Для великих проектів таке штучне заниження кількості переглядів призводить до значного падіння доходів.

Сайти з технологією AJAX можуть мати гіршу репутацію у пошукових систем в порівнянні з аналогічними ресурсами без її використання. Серед основних причин:

- ймовірність, що пошукові боти НЕ врахують весь контент;
- марний адресний рядок (у всіх сторінок однакова адреса);
- робот може бачити не той зміст сторінки, що відображається користувачеві.

Цих негативних сторін можна уникнути, якщо використовувати AJAX за цільовим призначенням - для динамічного взаємодії з сервером. Наприклад, частина статті з ключовими словами встановити не динамічно на початку сторінки.

Динамічні сторінки можна кешувати і відображати їх в якості статичних. Для виклику AJAX краще скористатися класичним якорем, ніж подією «onClick».

Таким чином, грамотне використання AJAX дозволить підвищити швидкість роботи сайту та його зручність для користувачів, не жертвуючи дружнім ставленням з боку пошукових систем.

2.2.5.Bootstrap

Bootstrap - це безкоштовна та відкрита корінь розробки для створення веб-сайтів та веб-програм. Функція Bootstrap побудована на HTML, CSS та JavaScript (JS), щоб полегшити розробку адаптивних, перших мобільних сайтів і додатків.

Bootstrap може бути згорнуто до трьох основних файлів:

- bootstrap.css - структура CSS;
- bootstrap.js - JavaScript / jQuery framework;
- шрифт (набір значків шрифту).

Крім того, для роботи Bootstrap потрібна функція jQuery. jQuery - це надзвичайно популярна і широко використовувана бібліотека JavaScript.

Реагентний дизайн дозволяє веб-сторінці виявити розмір та орієнтацію екрана відвідувача та автоматично адаптувати його відповідно.Перший підхід до мобільних пристроїв передбачає, що смартфони, планшети та мобільні додатки для конкретних завдань є основними інструментами працівників для виконання роботи та вирішення вимог цих технологій у дизайні.

Bootstrap включає в себе компоненти користувальницького інтерфейсу, макети та інструменти JS поряд з рамкою для реалізації. Програмне забезпечення доступне для попереднього копіювання або як вихідний код.

Марк Отто і Джейкоб Торнтон розробили Bootstrap на Twitter як засіб поліпшення узгодженості інструментів, що використовуються на сайті, і зменшення технічного обслуговування. Програмне забезпечення раніше називалося Blueprint Twitter, і його іноді називають Twitter Bootstrap.

У комп'ютерах слово bootstrap означає завантаження: завантажувати програму в комп'ютер за допомогою набагато меншої початкової програми для завантаження в потрібну програму (яка зазвичай є операційною системою).

2.2.6.Adobe Photoshop

Adobe Photoshop - це програмне забезпечення, яке широко використовується для редагування растрових зображень, графічного дизайну та цифрового мистецтва. Він використовує шаруваність, щоб забезпечити глибину та гнучкість у процесі проектування та редагування, а також

забезпечити потужні інструменти для редагування, які, коли поєднуються, здатні бути практично будь-якими.

Він був створений братами Томасом і Джон Ноллом у 1989 році. У 1989 році Джон продав програму Adobe Systems, яка продала її як "Photoshop". З того часу програма стала де-факто галузевим стандартом для редагування растрової графіки. Він публікується як для macOS, так і для Windows, але не для Linux.

Photoshop головним чином призначений для редагування цифрових фотографій та створення растрової графіки. Особливості Adobe Photoshop полягають у багатому інструментарії для операції створення і обробки зображень, високій якості обробки графічних зображень, зручності й простоті в експлуатації, широких можливостях до автоматизації обробки растрових зображень, які базуються на використанні сценаріїв, механізмах роботи з кольоровими профілями, які допускають їх втілення в файли зображень з метою автоматичної корекції кольорових параметрів при виводі на друк для різних пристроїв, великому наборі команд фільтрації, за допомогою яких можна створювати найрізноманітніші художні ефекти.

Для роботи з окремими фрагментами зображення передбачені різні типи виділення: за фігурою, в режимі «малювання» зони виділення, за діапазоном кольорів тощо. Існують різноманітні фільтри для деформації та стилізації зображення, такі як фільтри розмиття, імітації різних художніх технік. Photoshop також містить інструменти для цифрового живопису, зокрема набори пензлів. Користувач може змінювати їх розмір, кут нахилу, колір. Підтримується встановлення сторонніх пензлів, стилів, шрифтів, палітр. Попри те, що спочатку програма була розроблена як редактор зображень для поліграфії, в наш час вона широко використовується і у веб-дизайні. У більш ранній версії була включена спеціальна програма для цих цілей — Adobe ImageReady, яка була виключена з версії CS3 за рахунок інтеграції її функцій в сам Photoshop, а також включення в лінійку

програмних продуктів Adobe Fireworks, що перейшло у власність Adobe після придбання компанії Macromedia.

Photoshop тісно пов'язаний з іншими програмами для обробки медіафайлів, анімації та іншої творчості. Спільно з такими програмами, як Adobe ImageReady (програма скасована у версії CS3), Adobe Illustrator, Adobe Premiere, Adobe After Effects і Adobe Encore DVD, він може використовуватися для створення професійних DVD, забезпечує засоби нелінійного монтажу і створення таких спецефектів, як фони, текстури і т. д. для телебачення, кінематографу і всесвітньої павутини. Основний формат Photoshop, PSD, може бути експортований і імпортований всіма програмними продуктами, переліченими вище. Photoshop CS підтримує створення меню для DVD. Спільно з Adobe Encore DVD, Photoshop дозволяє створювати меню або кнопки DVD. Photoshop CS3 у версії Extended підтримує також роботу з тривимірними шарами.

Базові інструменти редагування дозволяють змінювати тон, насиченість зображення, обтинати його, накладати фотофільтри, виправляти перспективу тощо. Photoshop підтримує так звані шари — прозорі області зображення, на яких розміщуються елементи фотомонтажу, текст, геометричні фігури. Програма містить інструменти для роботи з текстом і нескладними фігурами, дозволяє малювати робочі контури, задавати текстам і фігурам стилі оформлення.

2.3.Структура та алгоритми створення серверної частини

Серверна частина веб – орієнтованої системи – це технологія, яка необхідна для обробки вхідного запиту, а також генерація та отправка відповіді клієнту через браузер. Як правило, серверна частина складається з трьох частин:

- Сервер - це просто комп'ютер, який слухає вхідні запити. Хоча існують і створені, і оптимізовані для цієї конкретної мети

машини, будь-який комп'ютер, підключений до мережі, може працювати як сервер.

– Додаток. Це додаток, що працює на сервері, який обробляє запити, витягує інформацію з бази даних і надсилає відповідь.
– База даних являє собою програмне забезпечення, яке використовується для організації та зберігання даних на сервері. В будь-який момент можна витягнути з бази даних потрібну інформацію за допомогою запиту. Зберігання даних в базі даних зменшує навантаження на основну пам'ять серверного ЦП і дозволяє отримувати дані, якщо сервер втрачає потужність.

Це основні компоненти більшості веб-орієнтованих систем. Схему їх взаємодії можно уявити таким чином, яка зображена на рис. 2.2.

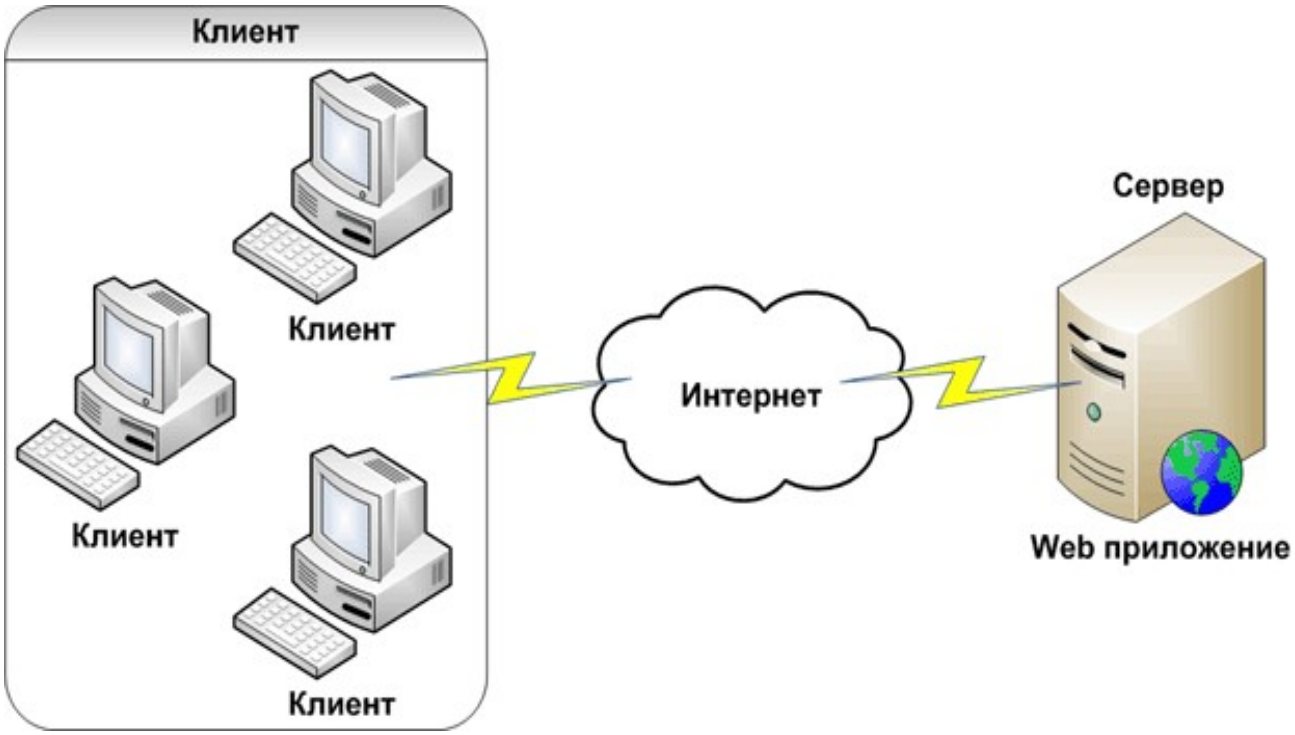

Рисунок 2.2 - Взаємодія компонентів веб-додатку

– Браузер відсилає запити HTTP веб-серверу через інтернет;
– Веб-сервер визиває скрипт, який написав веб-розробник;
– Скрипт робить запит до бази даних, якщо це необхідно.

– Скрипт вертає користувачу веб-сторінку, яку відображає браузер через інтернет.

Веб-орієнтовані системи або веб-додатки являють собою особливий тип програм, побудованих за архітектурою "клієнт-сервер". Особливість їх полягає в тому, що саме веб-додаток знаходиться і виконується на сервері - клієнт при цьому отримує тільки результати роботи. Робота програми ґрунтується на отриманні запитів від користувача (клієнта), їх обробці та видачі результату. Існує безліч різних видів клієнтів: вони можуть бути мобільною програмою, програмою, що працює на іншому сервері, або навіть смарт-пристрій з підтримкою Інтернету. Передача запитів і результатів їх обробки відбувається через Інтернет.

Відображенням результатів запитів, а також прийомом даних від клієнта і як вони будуть передані на сервер зазвичай займається спеціальний додаток - браузер (Internet Explorer, Mozilla, Opera і т. Д.). Як відомо, однією з функцій браузера є відображення даних, отриманих з Інтернету, у вигляді сторінки, описаної на мові HTML, отже, результат, який передається сервером клієнтові, повинен бути представлений на цій мові. На стороні сервера Web-додаток виконується спеціальним програмним забезпеченням (Web-сервером), який і приймає запити клієнтів, обробляє їх, формує відповідь у вигляді сторінки, описаної на мові HTML, і передає його клієнту. В процесі обробки запиту користувача Web-додаток компонує відповідь на основі виконання програмного коду, що працює на стороні сервера, Web-форми, сторінки HTML, інший вміст, включаючи графічні файли. В результаті, як вже було сказано, формується HTML-сторінка, яка і відправляється клієнту. Виходить, що результат роботи Web-орієнтованої системи ідентичний результату запиту до традиційного Web-сайту, проте, на відміну від нього, Web-додаток генерує HTML-код в залежності від запиту користувача, а не просто передає його клієнту в тому вигляді, в якому цей код зберігається в файлі на стороні сервера. Тобто Web-додаток динамічно формує відповідь за допомогою коду, - так званої виконуваної частини.

За рахунок наявності виконуваної частини, веб-додатки здатні виконувати практично ті ж операції, що і звичайні Windows-додатки, з тим лише обмеженням, що код виконується на сервері, в якості інтерфейсу системи виступає браузер, а в якості середовища, за допомогою якої відбувається обмін даними, - Інтернет. До найбільш типових обробок, які виконуються веб-системами відносяться:

- прийом даних від користувача і збереження їх на сервері;
- виконання різних дій за запитом користувача: вилучення даних з бази даних (БД), додавання, видалення, зміна даних в БД, проведення складних обчислень;
- автентифікація користувача і відображення інтерфейсу системи, яке відповідає даному користувачеві;
- відображення постійно змінюється оперативної інформації і т. д.

2.4. Технології для створення серверної частини

2.4.1.PHP

Для побудови серверної частини веб-орієнтованої системи використовується мова програмування PHP, сервер OpenServer, самописна система управління контентом (CMS BlizAnt) та система управління базою даних MySQL.

При аналізі вже запущених в експлуатацію систем журнал Spectrum склав наступну статистику використовуваних мов програмування на 2018 рік (рис. 2.3).

Довго аналізуючи, яку саме вибрати мову програмування для серверної частини, PHP став найкращим вибором, так як він найбільш відповідає економічним вимогам і вимогам по трудомісткості

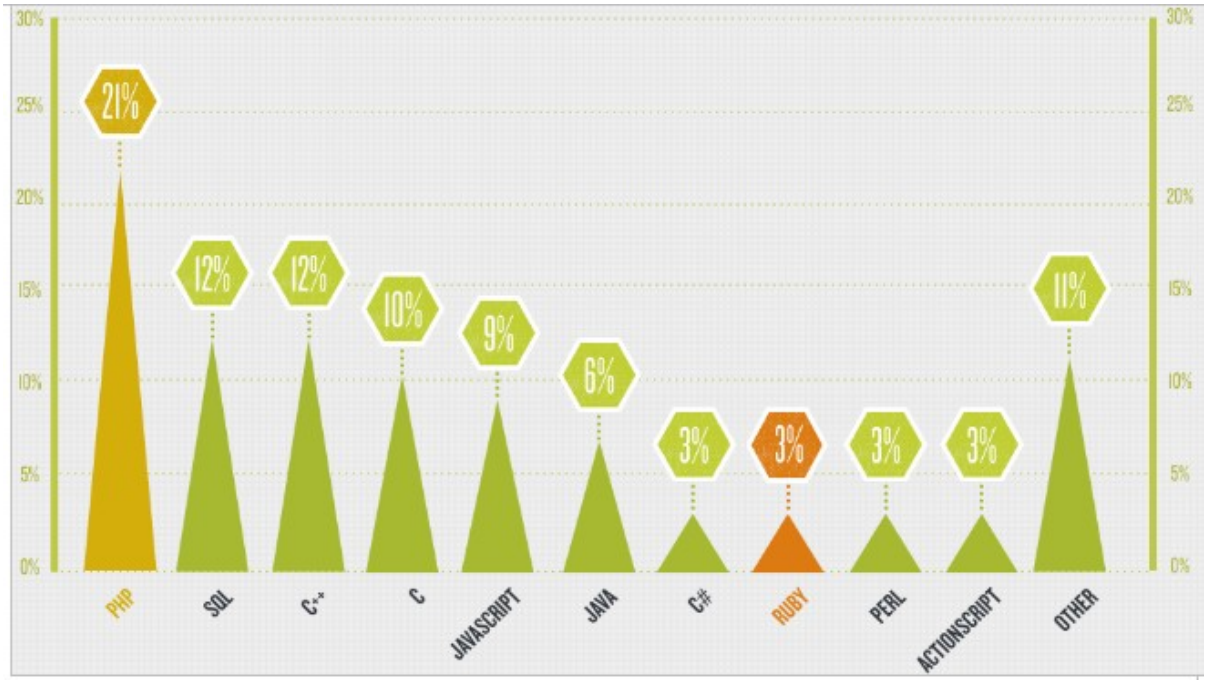

Рисунок 2.3 – Процентне співвідношення мов програмування

PHP - це мова програмування, призначена для створення сайтів. PHP дозволяє автоматизувати роботу з сайтом. Це скрипт-мова, вбудована в HTML, яка інтерпретується і виконується на сервері. Відмінність PHP від JavaScript, полягає в тому, що PHP-скрипт виконується на сервері, а клієнту передається результат роботи, тоді як в JavaScript-код повністю передається на клієнтську машину і тільки там виконується.

У минулому столітті, щоб створити сайт «на світовому рівні», досить було просто вміти працювати з HTML і володіти художнім смаком. Через деякий час вимоги ускладнилися: використання JavaScript і Dynamic HTML стало рутинним, а дизайн сторінок, які не мають подібних «прикрас», вважався застарілим. Незабаром правилом «хорошого тону» для корпоративного сайту стало застосування складних програм на Perl або C ++. Однак Web-дизайнери не бажали миритися з такою ситуацією. І на світ з'явився PHP - мова програмування, що володіє можливостями складних скриптових мов, але в той же час дивно простий і легкий у вивченні і застосуванні.

PHP є впроваджуємою мовою сценаріїв. Мета мови полягає в тому, щоб дозволити веб-розробникам швидко створювати сторінки, які динамічно генеруються. Крім того, розробникам веб-додатків немає необхідності говорити, що веб-сторінки - це не тільки текст і картинки. Гідний уваги сайт повинен підтримувати деякий рівень інтерактивності з користувачем: пошук інформації, продаж продуктів, конференції і тощо. PHP працює як частина веб-сервера. У цій мові немає строгої типізації даних і немає необхідності в діях по виділенню / звільненню пам'яті. Програми, написані на PHP, досить легко читаються. Написаний PHP - код легко візуально прочитати і зрозуміти.

На PHP можна зробити все: обробляти дані з форм, генерувати динамічні сторінки, одержувати і посилати сеанси - куки (cookies). Крім цього в PHP включена підтримка багатьох баз даних (databases), що робить написання веб-додатків з використанням БД до неможливості простим. Додатково до всього PHP розуміє протоколи IMAP, SNMP, NNTP, POP3 і навіть HTTP, а також має можливість працювати з сокетами (sockets) і спілкуватися по інших протоколах. PHP може серйозно полегшити роботу творця сайту. За допомогою команди include () можна автоматично вставляти один і той же фрагмент HTML-коду в безліч сторінок, просто помістивши в них цю команду з ім'ям файлу, що містить код загальної частини. В результаті для поновлення такої загальної частини буде достатньо відредагувати один файл - той, який її, власне, і містить, і відповідно зміняться всі інші сторінки. Також, при оновленні новин редагування доведеться піддавати лише сам їх текст, а не HTML-код головної сторінки, що і самому робити легше, і недосвідченому помічникові довірити можна.

Допомога веб-майстру і службі підтримки веб-ресурсу - це лише дуже мала частина функцій PHP. Ця мова дозволяє значно збагатити сайт величезною кількістю нових функцій. Так, за допомогою функції mail () легко забезпечити відправлення поштового повідомлення з будь-якого адресою зі сторінок сайту і тощо.

Переваги PHP:

- створення та налагодження скриптів PHP значно простіше, ніж налагодження та створення скриптів на інших мовах;
- так як PHP-команди просто вставляються в текст html-документа, відпадає необхідність в різних IDE (інтегроване середовище розробки);
- для вирішення різних специфічних завдань не потрібно писати і налагоджувати численні маленькі CGI-програми, що зводить до мінімуму час доступу до ваших сторінок, а також тривалість розробки сторінок і сайту в цілому.

PHP є найперспективнішим з мов програмування для Інтернету, частка його використання в порівнянні з іншими мовами швидко зростає. Його основні переваги: широка підтримка різних технологій, сумісність з серверами, базами даних, простота і безкоштовність.

2.4.2.Open Server

Open Server - це безкоштовна вільно поширювана програма для веб-розробників, що включає в себе пакет компонентів серверного програмного забезпечення. Open Server потрібен для створення та налагодження повноцінних сайтів на локальному комп'ютері. Простіше кажучи, за допомогою цієї програми можна зробити аналог Linux серверів під Windows, і без проблем запускати сайти, наприклад, написані на PHP.

Open Server дуже простий в експлуатації. Для запуску програми не потрібно вміти конфігурувати сервера Apache і Nginx або налаштовувати MySQL. Програма це робить автоматично. Більш того Відкритий Сервер реалізований як портативний додаток, який не потребує установки. Програма може працювати з флешки, HDD і CD / DVD диска. В останньому випадку програма буде створювати тимчасовий каталог з даними на диску або віртуальному пристрої. Є можливість управління через консоль і створення власних збірок програми.

При необхідності програма може працювати як сервер в локальних або глобальних мережах. У разі роботи як інтернет-сервер необхідний статичний IP адреса, а так само слід пам'ятати про можливі загрози та захист свого комп'ютера від шкідливих скриптів.

Основний набір компонентів:

- сервер Apache;
- DNS сервер Bind;
- HTTP сервер Nginx;
- бази даних MySQL, MariaDB, MongoDB, PostgreSQL;
- журнальованою сховище даних Redis;
- система кешування даних Memcached;
- FTP сервер FTP FileZilla;
- середовище для виконання PHP скриптів PHP 5.x;
- інтерпретатор мови PostScript Ghostscript;
- сервіс для роботи з електронною поштою Sendmail;
- велика кількість інших допоміжних утиліт, таких як HeidiSQL, Adminer, RockMongo PHPMyAdmin та інших.

Програма представлена в трьох варіантах Basic, Premium, Ultimate. Різниця між ними в кількості додаткових утиліт для роботи. У базовій версії присутній тільки основний набір для запуску сервера. У версії Premium включені основні серверні програми і додаткові утиліти Git, ImageMagick, MongoDB, Rockmongo, PostgreSQL і PhpPgAdmin. У версію Ultimate входить велика кількість додаткових вільно розповсюджуваних програм для роботи з графікою, різні текстові та HTML редактори, програми для роботи з Інтернет, менеджери закачувань, SCP клієнти і так далі.

Open Server може працювати в обмеженому режимі в ситуації, коли немає можливості вносити зміни в файл HOSTS при обмеженні прав доступу або блокуванням файрволом. В цьому випадку для сайтів буде доступний тільки одну адресу localhost 127.0.0.1.

2.4.3.MySQL

Система управління базами даних (MySQL) - це комплекс програмних засобів, необхідних для створення структури нової бази, її наповнення, редагування вмісту і відображення інформації.

Переваги MySQL:

- СУБД з відкритим кодом. Будь-який користувач може безкоштовно завантажити програму на сайті розробника. Однак для застосування СУБД в комерційному додатку необхідно придбати комерційну ліцензовану версію програми у компанії MySQL AB.
- Кроссплатформенная система. Її можна використовувати практично у всіх сучасних операційних системах.
- Має безліч програмних інтерфейсів (API), завдяки яким до бази даних можуть підключатися додатки, створені за допомогою багатьох мов програмування.
- Має відмінні технічні характеристики: многопоточність, розрахована на багато користувачів, швидкодія і тощо.
- Має розвинену систему забезпечення безпеки і розмежування доступу на основі привілеїв.

Реляційна база даних існує у вигляді таблиць, що мають свої імена. На перетині кожного стовпця і кожного рядка розташований одне значення. Приклад зображен в таблиці 2.1

Таблиця 2.1 - Приклад зображення таблиці

Id(ідентифікатор)	Name(ім’я)	Phone(телефон)	Address(адрес)
533	Крилов	0951092018	Вул. Рокотова

Рядки таблиці можуть зберігатися в довільній послідовності і не повинні повторюватися.

Кожен стовпець таблиці має ім'я і тип даних, якому відповідають всі значення в стовпці. У прикладі, стовпці з іменами id і phone - числові, а з іменами name і address- символьні.

У свою чергу, для зручності роботи з СУБД використовуються спеціальні веб-додатки, які дозволяють за допомогою графічного інтерфейсу виконувати адміністрування сервера баз даних, запускати спеціальні команди, а також працювати з контентом таблиць і баз даних - дії, які при відсутності веб-додатки підлягають виконанню засобами консолі. Приклади: phpMyAdmin використовується для адміністрування СУБД MySQL, pgAdmin - для PostgreSQL. Ці програми управління базами даних ви знайдете і в cPanel на нашому віртуальному хостингу.

BlizAnt - це спеціальна програма зі зручною оболонкою для створення сайту і управління його контентом.

BlizAnt - це сучасна платформа з відкритим вихідним кодом, яка безкоштовна і вільна до поширення. Движок цієї системи допускає підключення зовнішніх модулів, її функціональність в даний момент практично нічим не обмежена, що дозволяє використовувати BlizAnt для розробки інтернет-ресурсів практично будь-якого типу: від звичайних блогів до новинних порталів зі складною структурою.

Основна перевага BlizAnt полягає в тому, що він надзвичайно гнучкий у використанні. Ідеологія BlizAnt - це легке і максимально швидке програмне ядро, що дозволяє підключати до нього необмежену кількість додаткових модулів (плагінів) в залежності від завдань, які стоять перед розробником. Крім того, BlizAnt дозволяє підключати готові зовнішні дизайн-рішення, що дає можливість зробити свій сайт по-справжньому унікальним.

Відмінні особливості BlizAnt:

- Простота установки CMS і початкової настройки сайту.
- Підтримка трансляції RSS-каналів (за допомогою RSS дається короткий опис нової інформації, що з'явилася на сайті, в наш час

наявність підписки по RSS стало необхідною умовою для будь-якого сучасного сайту і тим більше блогу).

- Можливість реєстрації відвідувачів сайту, а також підтримка багатокористувацького (англ. Multiuser) режиму, при якому кожен зареєстрований користувач має можливість вести свій власний блог на сайті.
- Підтримка тегів, наявність яких також стало стандартом для утримання будь-якого сучасного сайту.
- Можливість коментування матеріалів сайту відвідувачами, а для адміністратора сайту - можливість адміністрування коментарів, фільтрації "спаму". У стандартну постановку CMS не входить підтримка власного форуму, проте при необхідності ви можете легко встановити додатковий модуль, який дозволить додати на сайт форум і налаштувати його.
- Підтримка медіаформатів (аудіо, відео та зображення), можливість завантаження їх на сайт і коректного відображення на його сторінках.
- Огранізация поштової подписки для відвідувачів сайту - будь-який зацікавився вмістом може залишити свій email і отримати на нього нові матеріали.
- Підтримка розширень стандартного функціоналу у вигляді додаткових модулів (плагінів).
- Підтримка дизайн-тем, що дозволяють легко змінювати як зовнішній вигляд сайту, так і способи виведення даних. Крім того, ви можете звернутися до професійних веб-дизайнерів, які за порівняно невелику винагороду зроблять для вашого сайту дійсно неповторний дизайн.

І це неповний перелік можливостей, які надаються при використанні CMS BlizAnt, а завдяки механізму підтримки додаткових плагінів цей список практично нічим не обмежений.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСОБУ МОНІТОРИНГУ

Архітектура засобу моніторингу у вигляді веб-орієнтованої системи була виконана на основі самописної платформи BlizAnt зі зручною оболонкою для створення веб-застосунків та сайтів і управління їхнім контентом. Вона інкапсулює обробку об'єктів додатків високого рівня, таких як, наприклад, запит і відповідь, що робить їх доступними для розробника.

Для того, щоб структурувати серверну частину веб-орієнтованої системи, були взяті принципи архітектурного рішення MVC.

Розглянемо основні компоненти серверної частини веб-додатку, що були розроблені.

Model (Модель) - містить бізнес-логіку програми та включає в себе методи вибірки (це можуть бути методи ORM), обробку (наприклад, правила перевірки) та надання конкретних даних, що часто робить його дуже товстим, що є цілком нормальним. Користувач не повинен взаємодіяти з моделью безпосередньо. Модель не повинна проводити генерацію коду HTML та іншого, який може відрізнятися залежно від необхідності користувача. Одна та сама модель, наприклад: модель автентифікації людини може використовуватись як в адміністративній, так і в користувацькій частині програми. Моделі, зазвичай, дуже товсті і містять в собі більшу частину коду, пов’язаную з обробкою даних.

View (Вид) - використовується для встановлення зовнішнього відображення даних, отриманих від контролера та моделі.

- Вид містить розмітку HTML та невеликі вставки PHP-коду для сканування, форматування та відображення даних.
- Не повинен вид мати прямий доступ до бази даних. Це має стосуватися моделі.
- Не повинен працювати з даними, отриманими за запитом користувача. Це завдання повинно виконуватись контролером.

- Можна безпосередньо звернутися до властивостей і методів контролера або моделей, щоб підготуватися до вихідних даних.
- Перегляди, як правило, поділяються на загальний шаблон, що містить розмітку, загальну для всіх сторінок (наприклад, заголовок та нижній колонтитул) та частини шаблону, які використовуються для відображення даних, що виводяться з моделі або форм введення даних для відображення.

Controller (Контролер) – це зв’язуюча ланка, що з’єднує моделі, представлення даних та другі компоненти у виробничій програмі. Контролер несе відповідальність за обробку запитів користувачів. Контролер не має містити в собі запити SQL до бази даних. Вони найкраще зберігаються в моделях. Контролер не має містити в собі HTML та іншу розмітку. Вона повинна міститись у видах (View). У добре розробленій програмі MVC контролери зазвичай дуже тонкі і містять в собі небагато рядків коду.

Взаємозв'язок компонентів зображений на рис. 3.1.

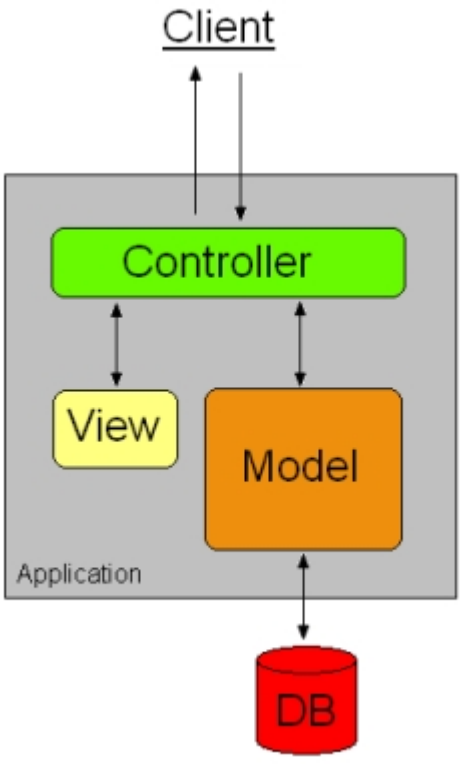

Рисунок 3.1 – Розробка веб-системи за принципом MVC

Веб-орієнтована система складається з чотирьох веб-сторінок та панелі адміністратора, яка запускається окремо від нашої системи. Перелік веб-сторінок системи:

- Головна сторінка;
- Законодавча база;
- Сторінка з початком консультації користувача;
- Новини;
- Сторінка з параметрами, по яким буде підбиратися відновлювальне джерело енергії;
- Сторінка з рекомендованим джерелом енергії, яке підібрала веб-орієнтована система

Кожна веб-сторінка розділена на три головні блоки:

- header (шапка системи) – це блок, який розташується у верхній частині і який користувач бачить на всіх веб-сторінках додатку. Як правило, він має в собі логотип, контакти, перемикач язиків (якщо сайт функціонує з двома мовами) .В моєму випадку, в блоці header знаходиться: логотип, перемикач язиків, кнопка авторизації та кнопка зареєструватися у системі, навігаційне меню.

```
<header>
……
<div class="bottom_header">
……
<ul>
        <li><a href=          "/data_base.php">Законодавча база</a></li>
</ul>
……
</div>
</header>
```

- content (middle_block) – це блок, в якому розташовується основна інформація, яка розміщена на сайті.

……

– footer (підвал сайту) – це блок, який розташується у нижній частині веб-сторінок. Він може зберігати в собі дані про автора, дата документу, дату створення ресурсу і тощо.

Структура головної сторінки функціональної частини розробленої веб-орієнтованої системи для моніторингу енергоефективних та відновлювальних джерел енергії зображена на рис. 3.2.

Рисунок 3.2 – Головна сторінка веб-орієнтованої системи

На головній сторінці сайту є кнопка реєстрації в системі. Реєстрація і авторизація дозволить клієнту отримати більше привілеїв, чим незареєстрованому користувачу.

Після натискання на кнопку реєстрації, з правої частини системи висувається popup (модальне вікно) з формою реєстрації. Воно зображено на рис. 3.3.

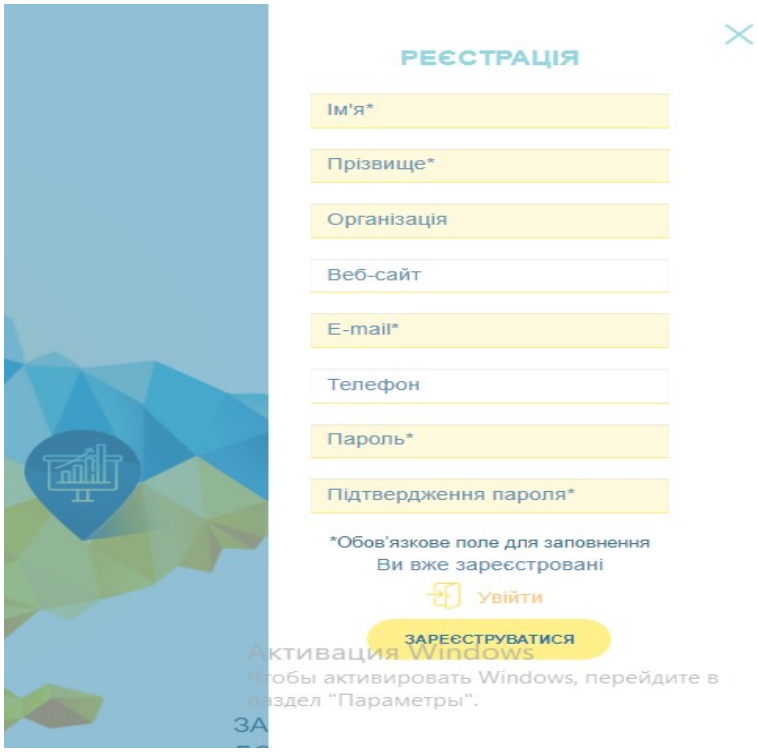

Рисунок 3.3 – Модальне вікно для реєстрації у системі

Функція відкриття модального вікна та його закриття реалізується за допомогою Javascript для реєстрації та авторизації.

```
$(document).on('click', '.open_popup', function () {
    var popup = $('.popup_overlay[data-popup=' + $(this).attr('data-popup') + ']');
    popup.fadeToggle(200);
```

}; - це функція з плавним відкриттям (продовження анімації 200 мілісекунд) вікна з реєстрацією. Кнопці реєстрації добавляємо клас .open_popup, а самій формі – клас .popup_overlay.

Аналогічно, записується функція для закриття модального вікна для двох випадків:

```
$(document).on('click', '.close_popup', function () {
```

```
        var popup = $('.popup_overlay[data-popup=' + $(this).attr('data-popup')
+ ']');
        popup.fadeToggle(200);
    };
```

Після появи popup з реєстрацією перед користувачем, він заповняє необхідні поля. Коли він все заповнив і натискає кнопку, то на стороні клієнта за допомогою JavaScript, перед відправкою і обробкою даних на сервері, відбувається валідація форми реєстрації. Якщо користувач пропустив поле або неправильно ввів дані, то під елементом форми змінюється колір лінії з жовтого на красний. Це можна побачити на рис. 3.4.

```
    function validate(Obj) {
        var ObjName = Obj.attr('name');
        var ObjVal = Obj.val();
        var wrongClass = 'has_er';
        var ObjType = Obj.attr('type');
        if (ObjName == 'email' || ObjName == 'mail') {
          if (isValidMail(ObjVal) && ObjVal != '') {
            Obj.closest('.control_wrapper').removeClass(wrongClass);
            return 0;
          } else {
            Obj.closest('.control_wrapper').addClass(wrongClass);
            return 1;
          }
        } else if (ObjName == 'upload') {
          return 0;
        } else if (ObjType == 'hidden') {
          return 0;
        } else {
          if (ObjVal != '') {
            Obj.closest('.control_wrapper').removeClass(wrongClass);
```

```
            return 0;
        } else {
            Obj.closest('.control_wrapper').addClass(wrongClass);
            return 1;
        }
    }
}
```

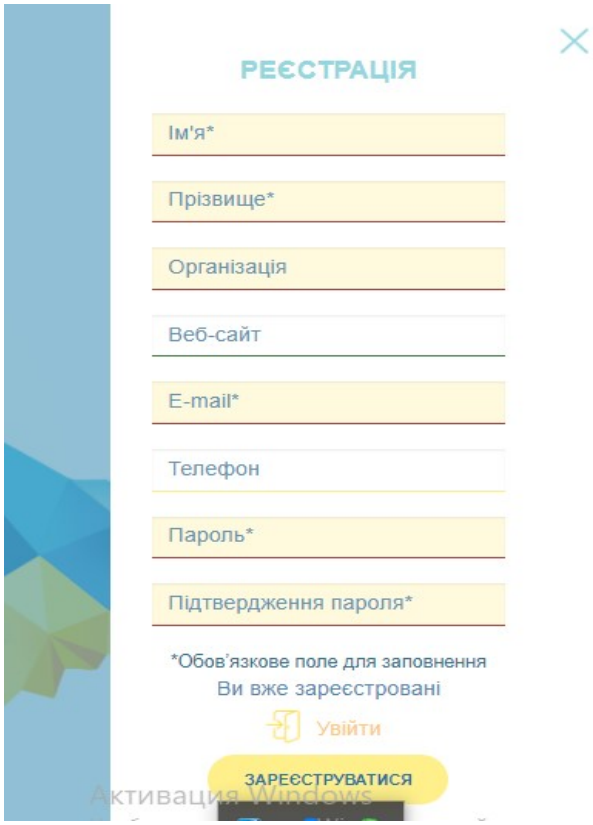

Рисунок 3.4 – Валідація полів форми реєстрації

Ці дві кнопки знаходяться не тільки на головній сторінці, а також на всіх інших і тому, щоб не повторювати і не вставлятий один й той же самий код на всіх сторінках, можна створити окремий файл з цими формами і підключити їх за допомогою PHP конструкції require_once './forms/registration.php' . Запит require_once () може використовуватися для включення php-файлу в інший, коли вам може знадобитися включити називаний файл більше одного разу. Якщо виявиться, що файл вже був

включений, скрипт виклику буде ігнорувати подальші включення. Таким же методом були підключені header та footer, так як вони теж повторюються на кожній сторінці.

Після входу в систему, користувач автоматично переходить до сторінки з консультацією. На ній підгружається сама інтерактивна карта для користувача. Вона реалізована за допомогою мови програмування JavaScript.

Для того, щоб додати карту Google Maps на веб-сторінку, потрібно получити спеціальний API ключ, який можна получити за цим посиланням - developers.google.com/maps/documentation/javascript/tutorial. Для безкоштовного використання Google Maps API дозволяє 25 000 запитів на добу, що цілком достатньо для більшості сайтів.

Функція, для реалізації карти, виглядає наступним чином:

```
function initMap(){
   var element = document.getElementById('map');
   var options = {
      zoom: 6,
      center : {lat: 48.2484507, lng: 30.6898306}
   };
   if(!element){
   } else {
   var myMap = new google.maps.Map(element, options);
   }
}
```

В даному прикладі я поставив збільшення в 6 разів (максимум 20 - дома, мінімум 1 - весь світ). Карта Центрована на карті України.

Також, треба виділити для карти блок та добавити стилі:

```
<div id="map">
……
</div>
```

В даному прикладі висота карти height: 100% та ширина width: 100% виставлена. В іншому випадку наша карта не відобразилася б.

Після того, як загрузилась карта, перед користувачем з’являється нове модальне вікно (popup) з проханням ввести місто, в якому він живе і де він хоче встановити відновлювальне джерело енергій. Це зображено на рис. 3.5.

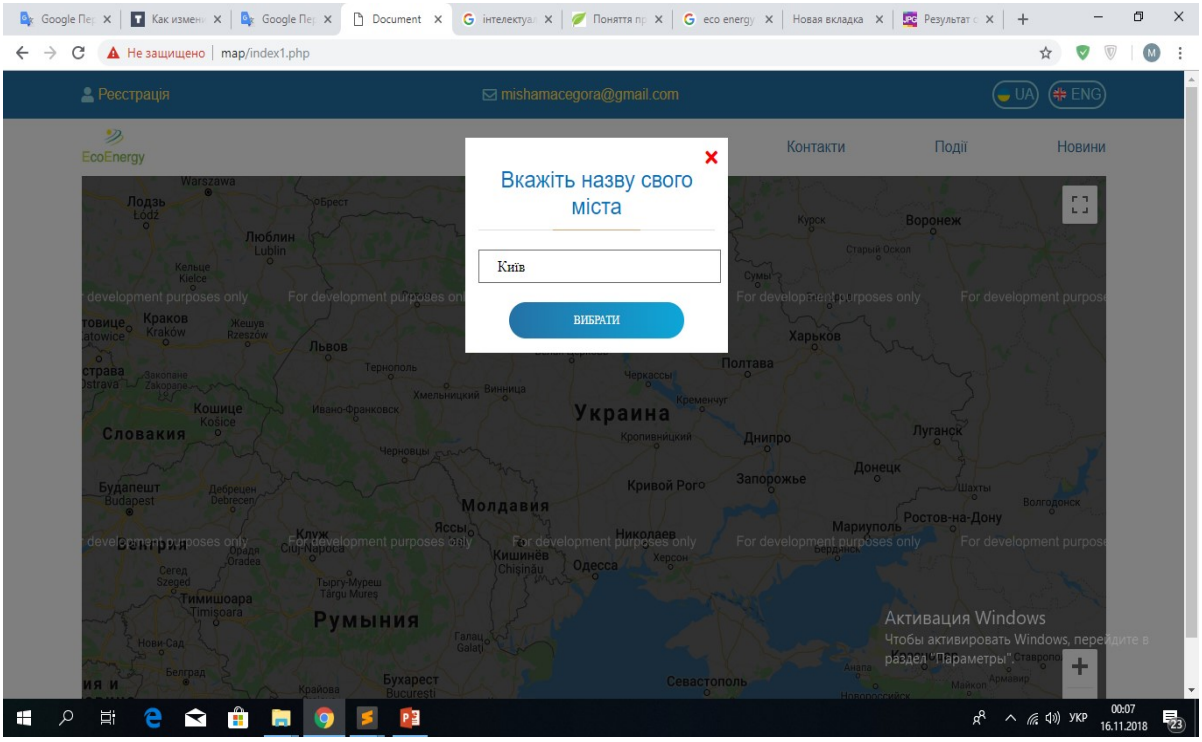

Рисунок 3.5 – Реалізація карти та вспливаюче модальне вікно

Після того, як користувач введе своє місто, в якому живе, тоді на Google Map появляється маркер з координатам та містом користувача.

Реалізується ця дія теж за допомогою Javascript. В функцію initMap додається кілька строк коду:

```
var marker = new google.maps.Marker({
        position: {lat: 50.42, lng: 30.55},
        map: myMap,
        title: 'Текст спливаючої підказки',
        icon: {
                url: "images/markers/svg/Arrow_3.svg",
```

```
                scaledSize: new google.maps.Size(64, 64)
        }
google.maps.event.addListener(marker, "click", function() {
                // infoWindow.open
                // Параметр map -  це змінна, яка містить об’єкт карти (оголошена на 8 строці)
                infoWindow.open(map, marker);
        });
google.maps.event.addListener(map, "click", function() {
                // infoWindow.close – закриваємо інформаційне вікно.
                infoWindow.close();
        });
});
```

Введене місто в формі відправляється на сервер за допомогою AJAX, потім в базі даних шукаються координати, яким відповідає дане місто і на карті встановлюється маркер. Результат цієї дії зображено на рис. 3.6.

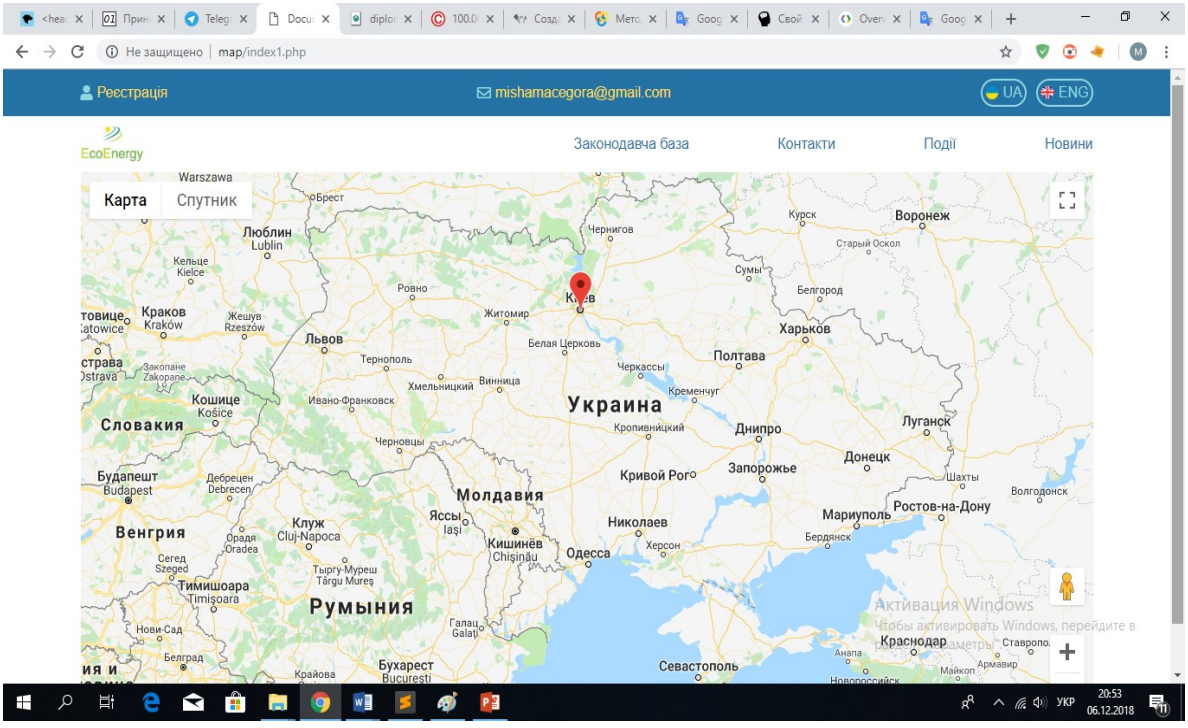

Рисунок 3.6 – Встановлення маркера на місто, в якому користувач буде встановлювати відновлювальне джерело енергії

Після того, як користувач наведе на маркер , через кілька секунд з’явиться спливаюча підказка. При натисканні на маркер відкривається інформаційне вікно з подальшою вказівкою по використанню (в даному випадку – це перехід до наступного етапу підбіра відновлювального джерела енергії) . Посилання виділено жовтим коліром.

Ця дія реалізона та показана користувачу на рис. 3.7.

Інформаційне вікно (balun) - це відмінний спосіб відображення інформації про певну точку або об'єкт.

Реалізується так само в функції initMap:

```
// Створюємо маркер
var marker = new GMarker(new GLatLng(56.317213,43.993976))
// Додаємо обробку подій натискання на маркер
GEvent.addListener(marker, 'click', function() {
// При кліці відкриваємо інформаційне вікно з такою html розміткою
marker.openInfoWindowHtml('
<div class="infowindow">
        <h3><strong>місто Київ</strong></h3>
        <p><a href="/poll.php"><strong>Продовжити підбір джерела</strong> <i
        class="fas fa-long-arrow-alt-right"></i></a></p>
</div>
');
});
// Додаємо маркер на карту
map.addOverlay(marker);
```

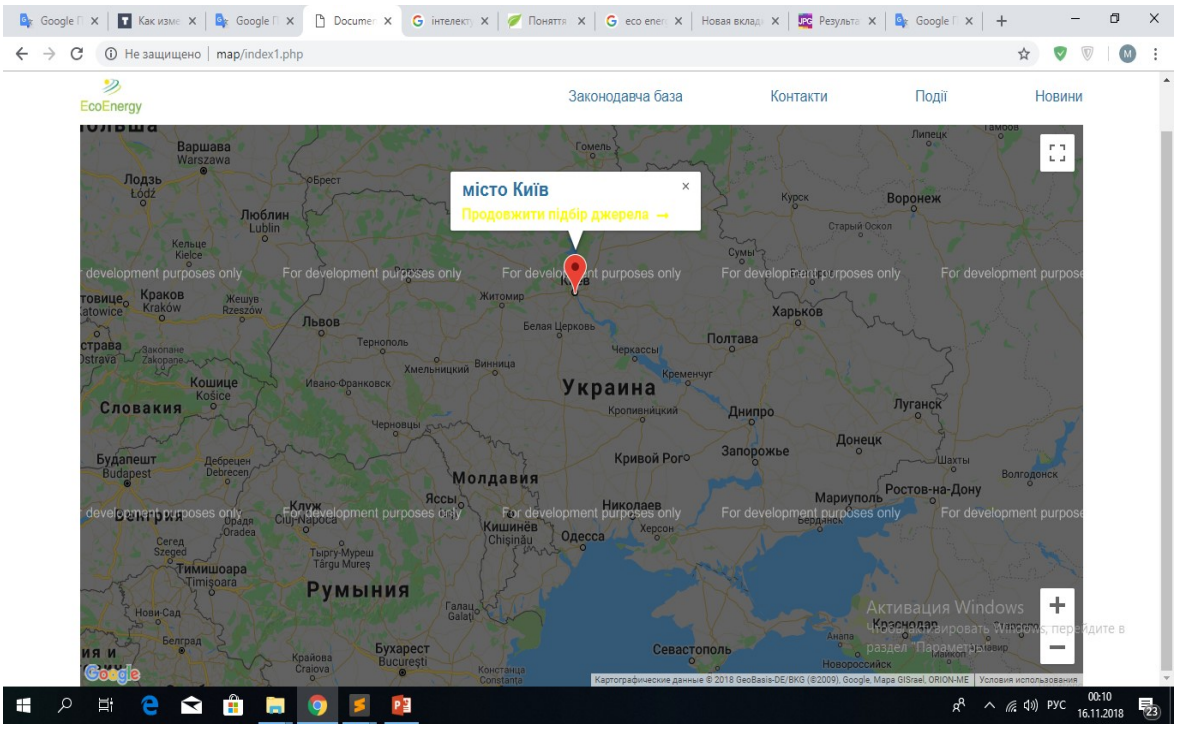

Рисунок 3.7 – Інформаційне вікно над маркером

Після переходу за посиланням, користувач переходить на сторінку з параметрами, по яким веб-орієнтована система буде підбирати відновлювальне джерело енергії. Все це можна побачити на рис. 3.8.

В даній системі джерело підбирається по чотирьом параметрам:

- Місце знаходження. Для всіх джерел енергії характерні залежності від місцезнаходження.
- Режим автономного енергозбереження. Є чотири види енергозбереження: повне, комфортне, помірне, базове та аварійне. Для кожного виду є пояснення.
- Тип будівлі.
- Площа використання.

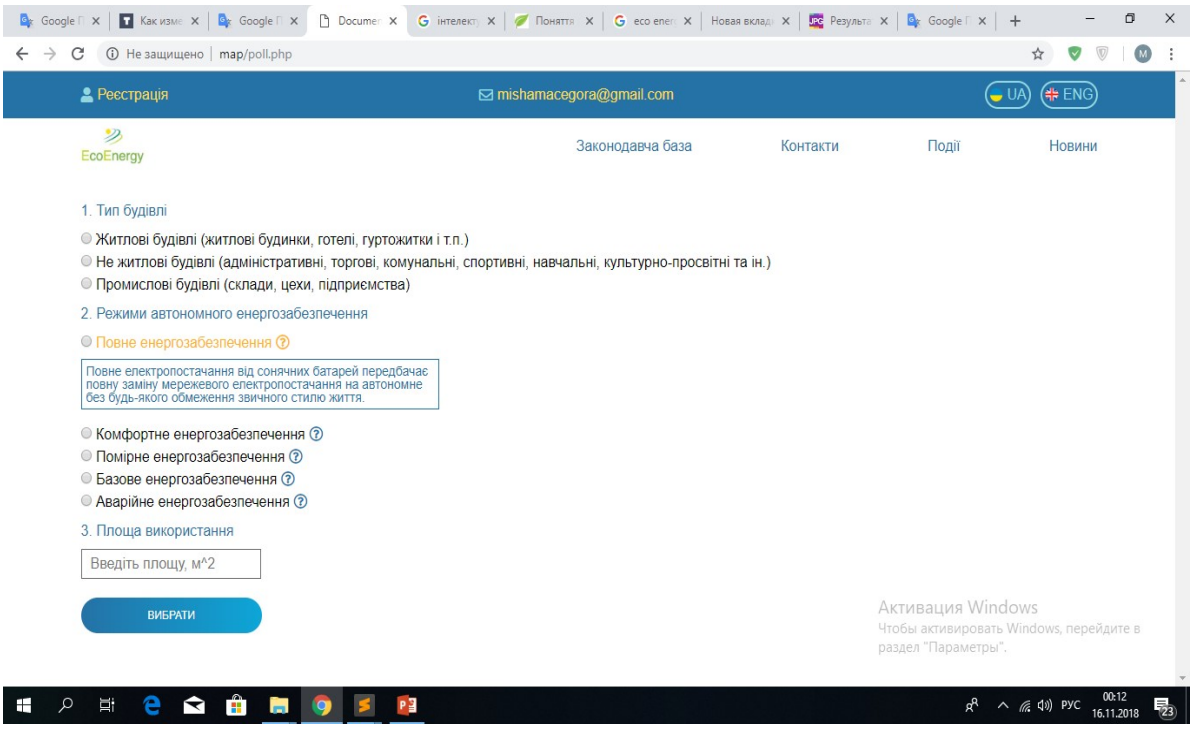

Рисунок 3.8 – Параметри, по яким підбирається відновлювальне джерело енергії

Коли користувач вибере всі параметри і натисне кнопку, то всі дані відправляться на сервер для аналіза і як тільки в БД знайдуться збіги якихось параметрів з характеристиками товарів, які лежать в БД і постійно поповнюються за допомогою ручного моніторингу різних ресурсів, то наш View (вид) виведе сторінку з рекомендованим джерелом енергії (рис. 3.9).

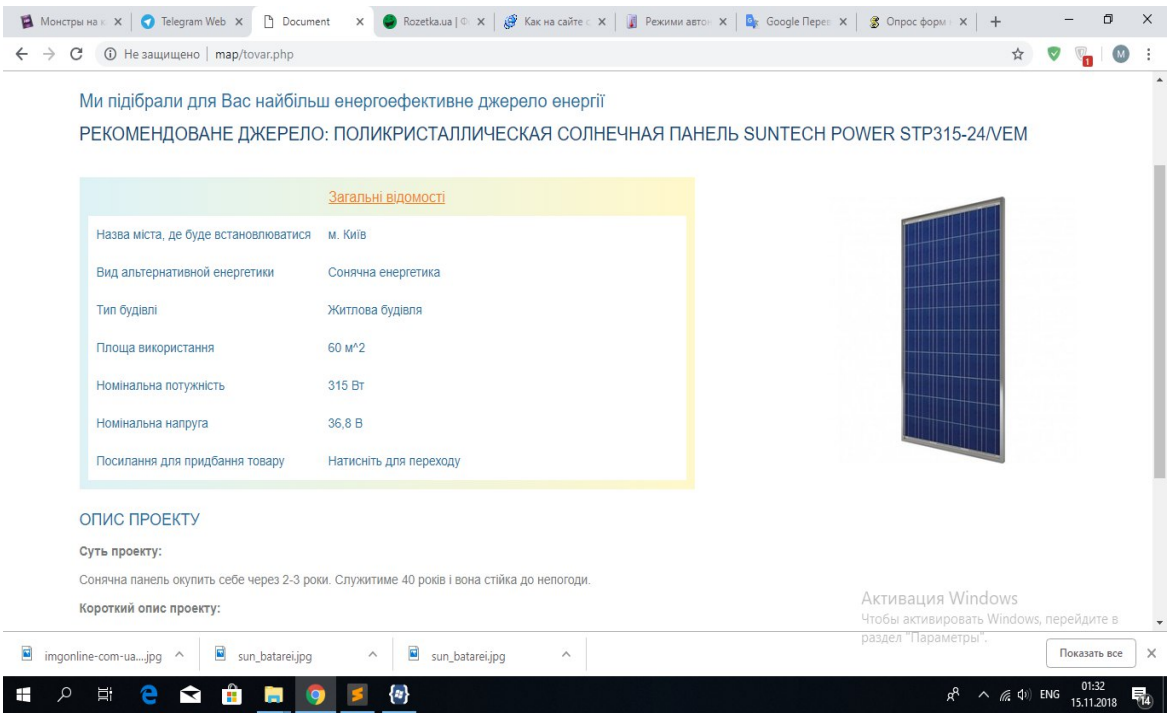

Рисунок 3.9 – Сторінка з рекомендованим джерелом енергії

ВИСНОВКИ

Для забезпечення інформацією споживачів альтернативної електроенергії, необхідно визначати, який саме продукт підходить під їх параметри. Для цих потреб у енергокомпаніях, за рахунок спеціалістів, створені спеціальні консультативні центри. Тому, можливе використання системи моніторингу.

В результаті виконання даної магістерської дисертації був створений засіб моніторингу у вигляді веб-орієнтованої системи з інтерактивною картою для енергоефективних та відновлювальних джерел енергії, який у реальному часі допоможе користувачу підібрати, яке джерело альтернативної енергії підійде саме для нього по його параметрам. Адже вибір певного джерела енергій залежить від багатьох факторів, які будуть враховуватись в даній веб-орієнтованій системі.

При використанні даного засобу моніторингу був відзначений ряд плюсів:

- Засіб дозволяє правильно підбирати джерело енергії під певні параметри;
- Максимально швидко та точно інформує про можливі варіації декілької джерел;
- Використання бази даних моніторингу для різних зовнішніх потреб;
- Простота використання.

В цілому запропонований засіб моніторингу може надавати якісну послугу клієнту, уникнувши при цьому грошових вкладень.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Обзор программ анализа и мониторинга сетевого трафика – [Електронний ресурс] – 2018 – Режим доступу: http://pi.314159.ru/volotka/volotka1.htm – Дата доступу: листопад 2018.

2. Кроудер, Дэвид. Создание веб-сайта для чайников, 3-е издание. Москва: издательство “Диалектика”, 2009.

3. Adobe Photoshop [Електронний ресурс] – 2006 – Режим доступу: https://uk.wikipedia.org/wiki/Adobe_Photoshop – Дата доступу: листопад 2018.

4. Структура Web-приложений – [Електронний ресурс] – 2018 – Режим доступу: https://www.intuit.ru/studies/courses/1139/250/lecture/6422– Дата доступу: листопад 2018.

5. AJAX – [Електронний ресурс] – 2018 – Режим доступу: https://learn.javascript.ru/ajax-intro – Дата доступу: листопад 2018.

6. Принцип MVC в web - программировании – [Електронний ресурс] – 2018 – Режим доступу: https://folkprog.net/printsip-mvc-u-web-programmirovanii/ – Дата доступу: листопад 2018.

7. Верстка веб-приложений – [Електронний ресурс] – 2018 – Режим доступу: https://semantica.in/blog/verstka-veb-sajtov.html – Дата доступу: листопад 2018.

8. Класифікація та принципи побудови систем моніторингу – [Електронний ресурс] – 2018 – Режим доступу: https://studfiles.net/preview/5706386/ – Дата доступу: листопад 2018.

9. Жуматий С.А., Кальянов А.А. Комплекс мониторинга распределенных информационно-вычислительных систем //

Научный сервис в сети Интернет: тр. Всерос. Науч. Конф. 2002. Изд-во МГУ.

10. Коваленко В., Коваленко Е. Пакетная обработка заданий в компьютерных сетях // Открытые системы. N7-8 2000.

11. Бараш Л. Мониторинг трафика в сетях с коммутацией пакетов / Бараш Л. // Компьютерное обозрение всех пакетов. – 2009. - № 37 (654). – С. 20-25